UNIVERSIDAD
COMPLUTENSE
MADRID

FACULTAD DE CIENCIAS ECONÓMICAS Y EMPRESARIALES

DOBLE GRADO EN Economía - Matemáticas y Estadística

TRABAJO DE FIN DE GRADO DE ECONOMÍA

TÍTULO: Optimización de carteras financieras bajo arquitecturas de redes neuronales recurrentes LSTM

AUTOR: Javier Molinero Araguas

TUTORA: Pilar Grau Carles

CURSO ACADÉMICO: 2025-2026

CONVOCATORIA: Febrero

Índice

Resumen

La inteligencia artificial ha adquirido un papel cada vez más relevante en las finanzas, especialmente en la predicción de variables de mercado. Se propone un modelo basado en redes neuronales recurrentes de tipo Long Short-Term Memory (LSTM) para la selección de activos financieros. La arquitectura desarrollada integra dos redes paralelas: una para estimar retornos esperados y otra para predecir volatilidades, entendidas como medida de riesgo. Ambas salidas se combinan mediante un meta-modelo inspirado en el Ratio de Sharpe, que genera señales de inversión y las traduce en pesos de cartera.

El modelo se entrena con datos históricos de múltiples activos y se valida mediante técnicas de backtesting en distintos escenarios de mercado. También se discuten limitaciones propias del enfoque, como el coste computacional y la escasa interpretabilidad, y se proponen líneas de mejora basadas en modelos híbridos y técnicas de inteligencia artificial explicable (XAI).

Palabras clave: Predicción financiera, Redes Neuronales Recurrentes (LSTM), Ratio de Sharpe, Optimización de carteras, Modelos GARCH, Backtesting, Inteligencia Artificial Explicable (XAI).

Abstract

Artificial intelligence has gained increasing relevance in finance, particularly in market variable forecasting. This work proposes a model based on Long Short-Term Memory (LSTM) recurrent neural networks for financial asset selection. The architecture integrates two parallel networks: one estimating expected returns and another predicting volatility as a proxy for risk. Both outputs are combined through a *Sharpe-Inspired* meta-model that generates investment signals and translates them into portfolio weights.

The model is trained on historical data from multiple assets and validated through backtesting across different market conditions. Limitations related to computational cost and model interpretability are discussed, and potential improvements are suggested through hybrid architectures and explainable artificial intelligence (XAI) techniques.

Keywords: Financial forecasting, Recurrent Neural Networks (LSTM), Sharpe Ratio, Portfolio Optimization, GARCH models, Backtesting, Explainable AI (XAI).

Sección 1. Introducción

1.1. Motivación

La inteligencia artificial (IA) se ha convertido en uno de los ámbitos de investigación más populares en los últimos años, hasta el punto de que es difícil encontrar un área aplicada que no esté siendo examinada bajo este prisma.

Las finanzas no son una excepción: el atractivo de anticipar el comportamiento de activos y mercados es enorme y alimenta de forma constante nuevas aproximaciones y experimentos. Ahora bien, "IA" es un término amplio bajo el que coexisten técnicas y objetivos distintos, como regresión para predecir valores, clasificación para asignar etiquetas o generación para crear nuevos contenidos, y el enfoque elegido condiciona de manera decisiva los resultados.

Por último, señalar que este proyecto encaja especialmente bien con la doble titulación, pues combina herramientas cuantitativas (álgebra lineal, estadística, probabilidad y series temporales) con fundamentos financieros y económicos (riesgo, diversificación, valoración, macro y microeconomía), haciendo del TFG un punto de encuentro natural entre ambos campos.

1.2. Objetivos y plan de trabajo

El objetivo fundamental del trabajo es ver cómo se desenvuelven modelos de aprendizaje automático en el campo de la predicción y decisión de valores financieros. Para este fin se construirá una cartera de valores elegida por un modelo basado en redes neuronales. La construcción de dicho modelo constituye el principal reto de este trabajo y para ello se precisa:

- Automatizar la gestión de datos, usando automatización a través de Python y Yahoo Finance, así como el preprocesamiento oportuno de los mismos.
- Modelizar el valor de un activo y cómo este se ve afectado por distintas variables.
- Incorporar gestión de riesgos, abordando aspectos como el riesgo sistemático o la diversificación de activos.
- Lograr la automatización plena del modelo haciendo que este sea completamente autónomo.

El plan de trabajo puede resumirse en cuatro etapas:

1. **Marco teórico**. Se reunirá bibliografía que sistematice las técnicas existentes en IA aplicada a las finanzas, así como conocer algunos modelos clásicos en optimización de carteras y predicción de riesgos.
2. **Recogida y preprocesamiento de datos**. Se trata de construir una base de datos de calidad, que permita entrenar y validar el modelo.
3. **Construcción de la arquitectura**. Se establecerá un sistema basado en redes neuronales. Esta parte del trabajo es la más técnica y extensa y se hará un código en Python que constituya el modelo y emita las predicciones.

4. **Valoración de los resultados**. Con los resultados del experimento, se comparará con los *benchmarks* elegidos para responder a la pregunta inicial acerca de cómo se desenvuelve el modelo, qué limitaciones presenta y en qué dirección se podría mejorar.

1.3. Estructura del documento

El trabajo se organiza en las siguientes secciones:

- **Sección 1: Introducción.** Presenta la motivación, los objetivos y la contribución del trabajo, así como una visión general del enfoque propuesto.
- **Sección 2: Marco teórico y fundamentos.** Revisa los conceptos financieros y metodológicos necesarios, y sitúa el problema en el contexto de la literatura.
- **Sección 3: Metodología y construcción del modelo.** Describe el diseño del sistema propuesto, el tratamiento de datos y la lógica de integración entre los modelos predictivos (LSTM) y el meta-modelo de decisión de cartera.
- **Sección 4: Resultados y evaluación.** Presenta los resultados del backtesting y el análisis del desempeño de las estrategias, así como métricas de riesgo y contraste de hipótesis relevantes.
- **Sección 5: Limitaciones y líneas de mejora.** Discute las principales limitaciones empíricas y metodológicas del enfoque y propone posibles extensiones.
- **Sección 6: Conclusiones.** Resume los hallazgos principales y extrae implicaciones desde el punto de vista técnico y económico.

Los aspectos más técnicos u otros relativos a las secciones se detallan en los apéndices siguientes:

- **Apéndice A**: Fundamentos y ecuaciones de LSTM.
- **Apéndice B**: Arquitectura, hiperparámetros y diagnóstico de desajustes.
- **Apéndice C**: Mecanismos financieros y reglas de construcción de cartera.
- **Apéndice D**: Métricas internas de validación de las redes.
- **Apéndice E**: Reconstrucción de precios a partir de retornos.
- **Apéndice F**: Escenario adicional con apalancamiento.
- **Apéndice G**: Listado de hiperparámetros.
- **Apéndice H**: Códigos utilizados.

Estos apéndices se incluyen para trazabilidad y reproducibilidad del trabajo, así como para ampliar información.

Sección 2. Marco teórico

2.1. Modelos tradicionales en optimización de carteras: El Modelo Media-Varianza de Markowitz

El modelo de media-varianza de Markowitz (1952) constituye el punto de partida de la teoría moderna de carteras. Su objetivo es encontrar una combinación óptima de activos que minimice la varianza (riesgo) para un determinado nivel de rentabilidad esperada. En esencia, es un **problema de optimización bajo supuestos bien conocidos**: linealidad en la combinación de activos, normalidad de los rendimientos y estabilidad temporal de la covarianza.

En términos matemáticos, sea un universo de n activos, con vector de rendimientos esperados $\boldsymbol{r} = (r_1, r_2, \ldots, r_n)$, matriz de covarianzas Σ, y vector de pesos de cartera $\mathbf{w} = (w_1, w_2, \ldots, w_n)$, donde $\sum_{i=1}^{n} w_i = 1$.

El rendimiento esperado de la cartera es:

$$E(R_p) = \mathbf{w}^t \boldsymbol{r}.$$

La varianza (riesgo) de la cartera se expresa como:

$$\sigma_p^2 = \mathbf{w}^t \Sigma \, \mathbf{w}.$$

El problema de optimización puede formularse como:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^t \Sigma \, \mathbf{w} \\ \text{s.a.} \quad & \mathbf{w}^t \boldsymbol{r} \geq r^*, \\ & \sum_{i=1}^{n} w_i = 1. \end{aligned}$$

donde r^* representa el rendimiento mínimo deseado.

La solución de este problema genera la denominada *frontera eficiente*, es decir, el conjunto de carteras óptimas en el espacio riesgo-rendimiento.

2.2. Modelos en series temporales (GARCH-EGARCH) para predicción de riesgos

De forma complementaria, los modelos GARCH y EGARCH, ampliamente utilizados en el estudio de la volatilidad, parten de un enfoque dinámico en el que la varianza condicional depende de la información pasada. En el caso del modelo GARCH(1,1), la varianza se actualiza según la expresión:

$$\sigma_t^2 = \omega + \alpha \varepsilon_{t-1}^2 + \beta \sigma_{t-1}^2,$$

donde la varianza presente se explica por los choques y volatilidades previas, incorporando así una estructura de memoria en el comportamiento de los rendimientos. No

obstante, este modelo asume que los choques positivos y negativos afectan de igual manera a la volatilidad, lo cual no siempre se cumple en los mercados financieros.

En mercados financieros, las caídas tienden a elevar más la volatilidad que subidas equivalentes, fenómeno conocido como **efecto apalancamiento** (*leverage effect* en inglés). Para capturar esta asimetría, Nelson (1991) propuso el modelo EGARCH, que introduce una formulación logarítmica de la varianza. En su forma básica EGARCH(1,1), la dinámica se expresa como:

$$\log(\sigma_t^2) = \omega + \beta \log(\sigma_{t-1}^2) + \alpha \left(\frac{|\varepsilon_{t-1}|}{\sigma_{t-1}} - E\left[\frac{|\varepsilon_{t-1}|}{\sigma_{t-1}} \right] \right) + \gamma \frac{\varepsilon_{t-1}}{\sigma_{t-1}},$$

donde el parámetro γ mide la asimetría: si $\gamma < 0$, los choques negativos incrementan más la volatilidad que los positivos. La formulación logarítmica, además, evita imponer restricciones de no negatividad sobre los parámetros, mejorando la estabilidad y flexibilidad del modelo.

Como señalan Casas and Cepeda (2008), la motivación fundamental de los modelos GARCH es capturar una propiedad esencial de las series financieras: la volatilidad no es constante, sino que tiende a agruparse en periodos de alta y baja intensidad, fenómeno conocido como *volatility clustering*. Los modelos GARCH de rendimientos financieros y sus extensiones, como el EGARCH, logran reproducir este comportamiento al hacer depender la varianza de los choques y varianzas previas, incorporando así memoria de los movimientos del mercado.

2.3. Introducción al *Machine Learning* y sus aplicaciones en finanzas

2.3.1. Tipos de modelos de aprendizaje

Un modelo de *machine learning* puede entenderse como un algoritmo que se entrena para alcanzar un objetivo concreto, y tanto el objetivo como la forma de ajuste (entrenamiento) determinan su desempeño.

Según cómo aprende a partir de los datos, se distinguen tres enfoques principales: **aprendizaje supervisado**, cuando se entrena con datos etiquetados; **no supervisado**, cuando descubre patrones sin etiquetas (por ejemplo k-means o PCA); y **por refuerzo**, cuando aprende interactuando con un entorno mediante recompensas y penalizaciones, especialmente útil en decisiones secuenciales como la gestión de carteras.

En términos generales, cada enfoque tiene ventajas y limitaciones, y su utilidad depende del problema financiero que se quiera abordar.

2.4. Redes Neuronales Artificiales (RNA)

Dentro del extenso mundo del *machine learning*, este trabajo se centrará en las redes neuronales. En particular, se recurrirá a un tipo muy concreto llamado ***Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNN)***, es decir, redes

neuronales recurrentes de memoria a corto-largo plazo. Para entender por qué estas redes son idóneas en series temporales financieras, es necesario entender primero las bases teóricas que sustentan estas estructuras.

A continuación, se presenta una introducción teórica resumida a las redes neuronales. El Apéndice A recoge las ecuaciones de retropropagación y de la celda LSTM y la interpretación geométrica de las redes neuronales.

Como expone Bishop (2006), las redes neuronales constituyen una de las herramientas más potentes dentro del aprendizaje automático. Su objetivo es aproximar funciones complejas que relacionan un conjunto de variables de entrada con una o varias salidas, permitiendo capturar patrones no lineales que los modelos tradicionales difícilmente podrían modelar. En esencia, una red neuronal trata de aprender una función del tipo

$$\mathbf{y} = f(\mathbf{x}; \mathbf{w}),$$

donde $\mathbf{x}$ representa el vector de variables de entrada, $\mathbf{y}$ el de salida, y $\mathbf{w}$ el conjunto de parámetros (pesos y sesgos) que la red ajusta durante su entrenamiento. El propósito es que la red aprenda una función $f(\cdot)$ que se aproxime lo máximo posible a la generadora real de los datos.

2.4.1. Estructura general

Una red neuronal está formada por capas de neuronas conectadas entre sí. En las denominadas *feed-forward networks*, las redes clásicas, la información fluye siempre en un único sentido, desde la capa de entrada hasta la capa de salida, pasando por una o varias capas ocultas. La figura 1 muestra un esquema del flujo en este tipo de redes. Aquí, cada neurona (o nodo) realiza dos operaciones fundamentales: una combinación lineal de las entradas y una transformación no lineal de dicha combinación.

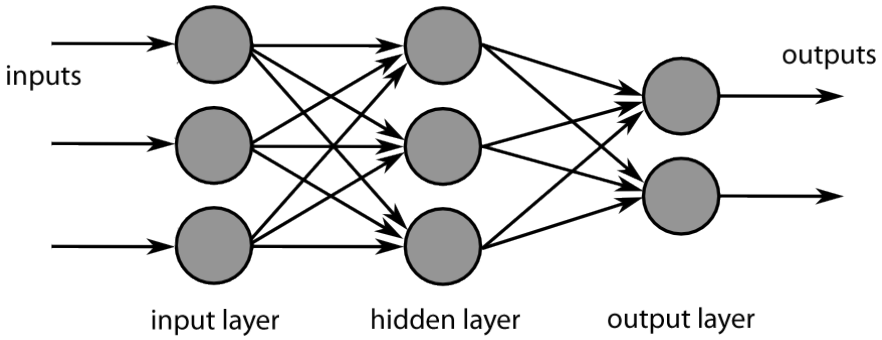

Figura 1: Estructura básica de una red neuronal "feed-forward". Fuente: Chrislb (2025)

Formalmente, para una capa oculta j se define una activación

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)},$$

y una salida

$$z_j = h(a_j),$$

donde $h(\cdot)$ es una función de activación que introduce la no linealidad (por ejemplo, la tangente hiperbólica o la sigmoide logística). A continuación, las salidas de esta capa se combinan en la capa siguiente:

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)},$$

$$y_k = \sigma(a_k),$$

siendo $\sigma(\cdot)$ la función de activación de la capa de salida (que puede ser la identidad para problemas de regresión o una softmax para clasificación).

En términos generales, la red puede expresarse de forma compacta como

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)}\, h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right).$$

Este proceso, conocido como *propagación hacia adelante* (*forward propagation*), traduce las entradas en una predicción de salida a través de transformaciones sucesivas.

En definitiva, cada neurona ajusta sus parámetros para minimizar su contribución al error global. Las primeras capas aprenden representaciones simples de los datos (por ejemplo, tendencias o patrones locales), mientras que las capas más profundas combinan estas representaciones en estructuras de mayor nivel. Esta jerarquía es precisamente lo que dota a las redes neuronales de su enorme capacidad de adaptación a diferentes conjuntos de datos.

2.5. Redes Neuronales Recurrentes y LSTM

Como describen Yu et al. (2019), las redes neuronales recurrentes (RNN, por sus siglas en inglés) surgen como una extensión natural de las redes neuronales tradicionales cuando el problema a resolver implica datos secuenciales. En esencia, una RNN introduce el concepto de memoria, permitiendo que la salida de una neurona dependa no sólo de la entrada actual, sino también del estado anterior del sistema. En otras palabras, las RNN están diseñadas para captar dependencias temporales, lo que las convierte en modelos idóneos para el tratamiento de series temporales, textos, audio o cualquier proceso donde el orden de los datos sea relevante.

2.5.1. Estructura básica de una RNN

A diferencia de una red neuronal feed-forward, donde la información fluye en una única dirección, en una red recurrente existe una conexión que retroalimenta las neuronas consigo mismas. Matemáticamente, esto se representa como:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b),$$
$$y_t = f(h_t),$$

donde x_t es la entrada en el instante t, h_t es el estado oculto (la "memoria" de la red), y y_t es la salida. Los pesos W_h y W_x determinan cómo la red combina la información nueva con la anterior, y $\sigma(\cdot)$ es una función de activación, típicamente la tangente hiperbólica o la sigmoide logística.

Esta estructura permite que el modelo "recuerde" información del pasado, pero con una limitación importante: cuando las secuencias son largas, la influencia de los datos antiguos se desvanece o, en algunos casos, se amplifica en exceso (*vanishing gradient problem* o *exploding gradient problem*) Bengio et al. (1994).

2.5.2. Motivación de las LSTM

Para superar este problema, Hochreiter and Schmidhuber (1997) propusieron la arquitectura *Long Short-Term Memory* (LSTM), que se traduce literalmente como "memoria a largo y corto plazo". La idea central es permitir que la red decida de forma autónoma qué información mantener, qué olvidar y qué actualizar en cada paso temporal. Esta decisión se toma mediante un conjunto de compuertas (*gates*) que regulan el flujo de información dentro de cada célula recurrente.

2.5.3. La célula LSTM

Puede interpretarse la LSTM como una neurona "inteligente", capaz de gestionar su propia memoria interna. En lugar de un simple estado oculto h_t, introduce una nueva variable, el *estado de celda* c_t, que actúa como una especie de cinta transportadora donde fluye la información relevante a lo largo del tiempo. Este estado se actualiza mediante tres compuertas fundamentales:

- **Puerta de olvido** (f_t): decide qué información del pasado debe olvidarse.
- **Puerta de entrada** (i_t): determina qué nueva información debe almacenarse.
- **Puerta de salida** (o_t): controla qué parte de la información interna se transmite al siguiente estado.

En la figura 2 se representa un esquema conceptual de una célula LSTM.

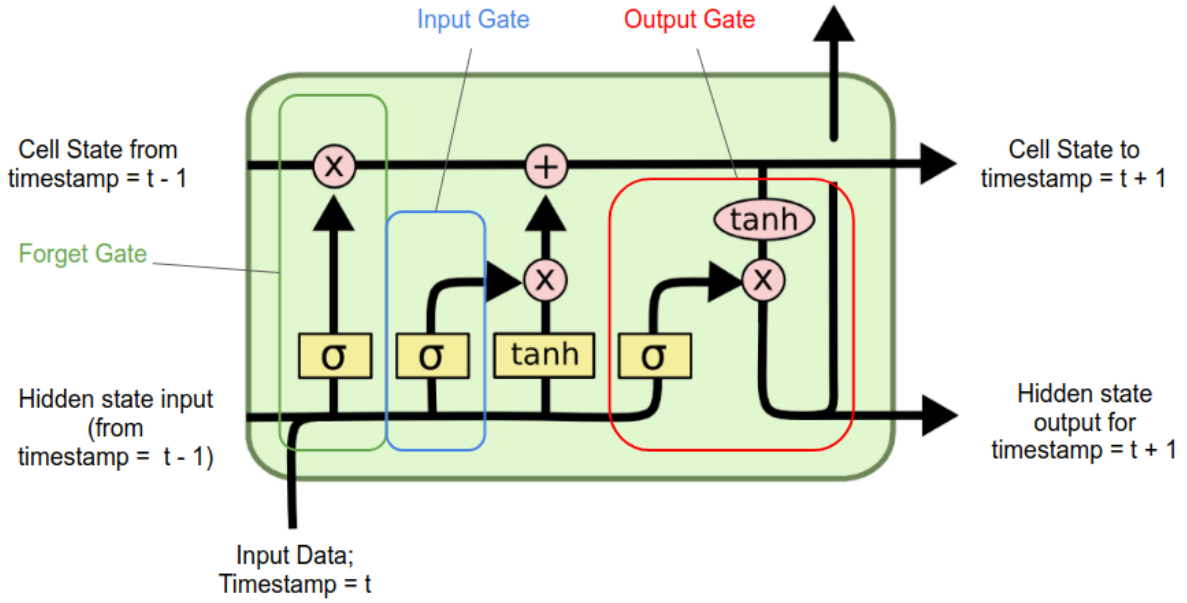

Figura 2: Estructura conceptual de una red "Long Short-Term Memory" (LSTM). Fuente: Dash et al. (2024)

Gracias a este diseño, la LSTM logra mantener dependencias a largo plazo sin que el gradiente desaparezca, lo que supone un avance crucial respecto a las RNN tradicionales.

En definitiva, las LSTM han demostrado un buen rendimiento en tareas secuenciales y, en el contexto financiero, pueden ser especialmente útiles para anticipar retornos o volatilidades al capturar dependencias temporales de distinta longitud mediante un mecanismo de compuertas que regula de forma adaptativa qué información conservar u olvidar. El mecanismo de compuertas actúa como memoria selectiva, preservando parte de la información pasada relevante para predicción. No obstante, estos modelos presentan limitaciones técnicas relevantes, como el riesgo de sobreajuste (Zhang and Zohren (2025)), además de un coste computacional elevado y la necesidad de ajustar numerosos hiperparámetros, aspectos que se analizarán en la sección de limitaciones.

2.6. Comparativa entre los modelos clásicos y las LSTM-RNN

Una vez presentados algunos enfoques clásicos de optimización de carteras y predicción de variables financieras, así como ciertas técnicas de aprendizaje profundo, resulta útil sintetizar una diferencia fundamental: mientras los modelos tradicionales se apoyan en formulaciones analíticas y supuestos estadísticos (como normalidad de rendimientos, estabilidad de correlaciones y estimación precisa de rendimientos esperados y covarianzas), las redes neuronales, particularmente las LSTM, aprenden directamente de los datos, capturando relaciones no lineales y patrones temporales sin imponer a priori una estructura paramétrica rígida.

En este sentido, los modelos tipo GARCH/EGARCH incorporan dinámica temporal mediante un número finito de retardos, pero presentan limitaciones para modelar dependencias complejas de largo plazo, mientras que las LSTM integran un mecanismo de memoria que ajusta de forma adaptativa qué información del pasado conservar, lo que las

hace más flexibles ante regímenes cambiantes.

No obstante, esta flexibilidad conlleva costes: mayor necesidad de datos, calibración más compleja y menor interpretabilidad, frente a la lectura económica inmediata de los parámetros en modelos clásicos, especialmente valiosa en análisis de riesgo.

En consecuencia, ambos enfoques deben entenderse como complementarios, y su integración puede dar lugar a modelos híbridos más robustos, como el propuesto por Zhou et al. (2021), que combina estructura económica y capacidad predictiva. Una vez establecidos estos fundamentos, se continúa con el desarrollo práctico del modelo.

Sección 3. La construcción del modelo

A continuación se describe el diseño de la arquitectura del modelo. En términos generales, los modelos de *machine learning* siguen un proceso metodológico que incluye la recopilación y el preprocesamiento de los datos, la partición del conjunto de datos en muestras de entrenamiento, validación y prueba, el entrenamiento del modelo, su evaluación sobre el conjunto de prueba mediante métricas adecuadas y el ajuste de hiperparámetros. Este procedimiento puede repetirse hasta alcanzar un desempeño satisfactorio.

El modelo sigue este esquema general, aunque con una arquitectura más elaborada. En particular, la arquitectura propuesta en este trabajo se basa en dos redes principales. La primera LSTM observa una ventana temporal de longitud L para los retornos de un activo y emite una predicción para un horizonte de dimensión h. La segunda red, por su parte, sigue la misma lógica pero trata de predecir la volatilidad, entendida como *proxy* del riesgo, de un activo.

Estas dos redes a nivel individual se basan en la metodología clásica de los modelos de *machine learning*, completando cada una de las etapas mencionadas hasta ser utilizable. Sin embargo, estas predicciones deben transformarse luego en pesos de cartera. Por ello se define un meta-modelo global que exige una validación distinta mediante la prueba de *backtest*. El sistema toma las predicciones de las dos LSTM y las procesa en una señal siguiendo un *modelo inspirado en Sharpe*. A partir de esa señal, se generan los pesos finales siguiendo una estrategia definida, que es la que se ejecuta en el mercado.

Ambas redes constituyen arquitecturas independientes, cada una con sus fases metodológicas propias. No obstante, el objetivo principal de este trabajo es el diseño de un meta-modelo financiero, que se apoya en las salidas de dichas redes pero no se limita a ellas. Este constituye el núcleo del análisis desarrollado y articula la integración de la información proporcionada por las dos arquitecturas neuronales.

En esta sección, se describe la metodología utilizada, y se expone cuál es el flujo del sistema desde que obtiene los datos hasta que completa la última transacción. Esta sección representa, por tanto, la transición del marco teórico al diseño práctico del sistema de predicción y selección de activos. La figura 3 resume el flujo completo del modelo.

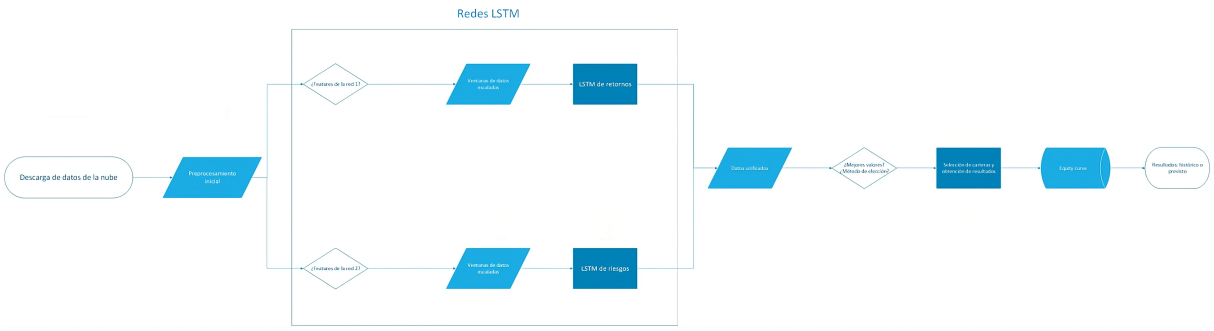

Figura 3: Diagrama de flujo del modelo completo. Fuente: Elaboración propia.

3.1. Descarga de datos, preprocesamiento y *feature engineering*

El modelo parte de una base de datos limpia y reproducible a partir de precios de cierre ajustados de una serie de activos representativos de renta variable, combinando ETFs sectoriales de Estados Unidos (XLK, XLF, XLY, XLE, XLV, XLI y XLB) con cuatro *blue chips* europeas (SAN.MC, ENEL.MI, SIE.DE y NOVN.SW), para combinar diversidad sectorial y geográfica sin perder representatividad económica.

La principal limitación práctica es el coste computacional, lo que motiva el hecho de trabajar con un universo reducido. Aunque universos más amplios suelen mejorar el rendimiento (Fischer and Krauss (2018) emplea 500 acciones), aquí se adopta un enfoque con un conjunto pequeño, en línea con Chen et al. (2015).

Los datos se obtienen de *Yahoo Finance* usando precios de cierre ajustados (dividendos y *splits*) desde el 01/01/2013 hasta la última fecha disponible en la ejecución (12/11/2025), evitando incluir observaciones demasiado antiguas que podrían inducir patrones obsoletos, criterio también respaldado por Chen et al. (2015) frente a ventanas más largas como la de Fischer and Krauss (2018).

Tras la descarga, se visualizan las series para evaluar su comportamiento general (figura 4) y detectar anomalías tempranas para tomar decisiones de procesamiento.

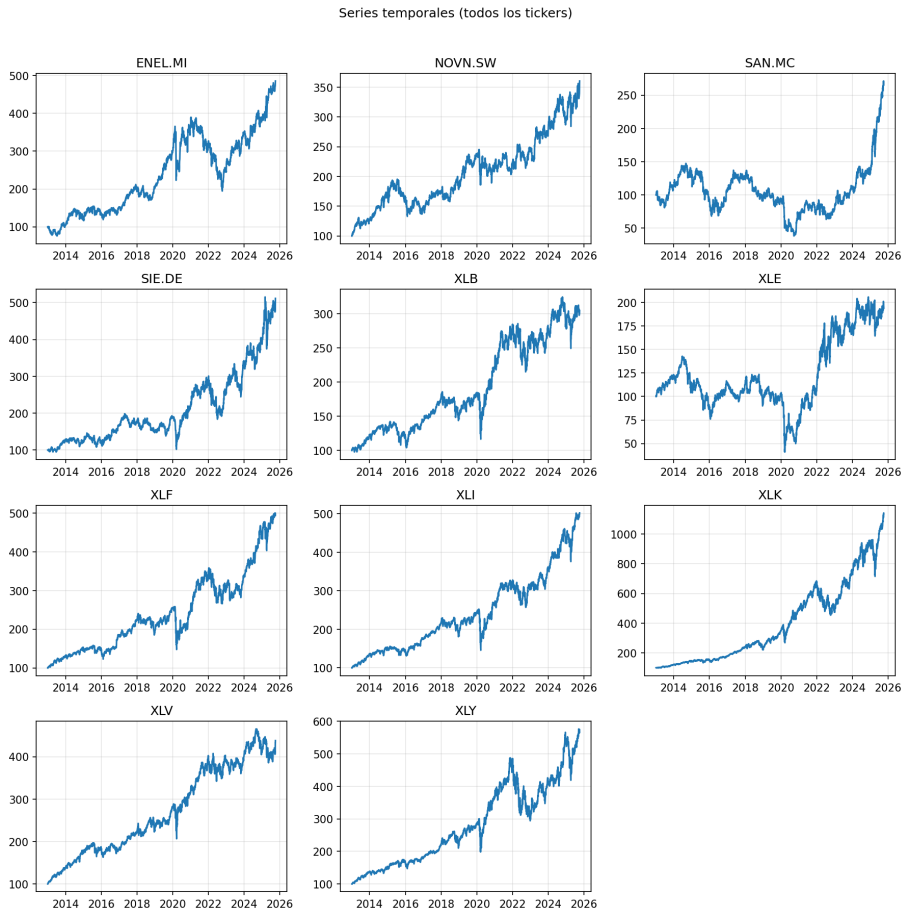

Figura 4: Evolución de los diferentes activos con base (01/01/2013=100). Fuente: Elaboración propia.

Una vez entendida la naturaleza de los datos, se realiza un proceso de limpieza y *feature engineering* de los mismos. Homogeneizar el conjunto de datos es necesario para el entrenamiento, pero además las *features* que contenga afectarán directamente al desempeño del mismo. Por *feature engineering* se entiende el conjunto de procedimientos y transformaciones aplicados a los datos en bruto para convertirlos en variables predictivas de calidad. El objetivo es maximizar el rendimiento del modelo combinando rigor estadístico y metodológico. En este trabajo, las decisiones se evalúan y se revisan si no contribuyen al desempeño. El proceso seguido se resume en:

1. **Limpieza y control de calidad.** Una vez obtenidas las series, se garantizan dos condiciones: orden cronológico estricto del índice temporal y consistencia multiactivo, eliminándose las observaciones con valores faltantes para homogeneizar geografías y eliminar efectos de calendario, evitando problemas durante el entrenamiento (el modelo no admite valores ausentes). Se eliminó el 5 % de las observaciones, aunque se conserva un tamaño muestral suficiente.

2. **Transformación a rendimientos.** Siguiendo la metodología habitual, se transforma la serie de precios en tasas de variación para lograr estacionariedad y una escala común. Se emplean rendimientos logarítmicos diarios por su aditividad temporal y buena aproximación a rendimientos continuos. Para cada activo i en el día t,

$$r_{t,i} \;=\; \log\!\left(\frac{P_{t,i}}{P_{t-1,i}}\right),$$

   donde $P_{t,i}$ es el precio ajustado.

3. **Tratamiento robusto de atípicos.** Para reducir la influencia de colas extremas, que pueden distorsionar tanto el entrenamiento como la estimación de covarianzas, se aplica una *winsorización* por columnas al 0,5 % y 99,5 %. Denotándose como Q_i^{α} el α-cuantil de $\{r_{t,i}\}_t$, la transformación es

$$\tilde{r}_{t,i} \;=\; \text{mín}\Big\{\,\text{máx}\{r_{t,i},\, Q_i^{0{,}005}\},\, Q_i^{0{,}995}\Big\}.$$

   Este procedimiento conserva la estructura temporal y las relaciones cruzadas, acotando únicamente los valores más extremos de cada serie. Con esta decisión se gana precisión en los valores centrales (momentos regulares del mercado) a costa de perder información en colas, en un *trade-off* de robustez.

   A modo de ejemplo, el tratamiento que hace la *winsorización* se puede apreciar en la figura 5, donde se aplica a la serie de retornos del Banco Santander:

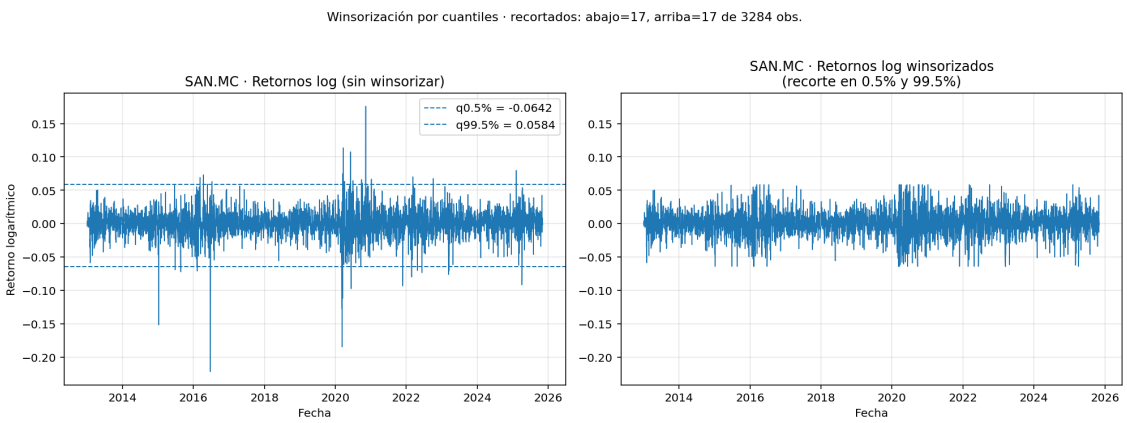

Figura 5: Winsorización al 0,5 % sobre SAN.MC. Fuente: Elaboración propia.

4. **Artefactos derivados.** Guardamos los datos en un fichero para su posterior uso, así como las matrices de covarianzas y correlaciones $\mathrm{Corr}(\tilde{r}_t)$, que se muestra en la figura 6.

$$\Sigma = \mathbb{E}\Big[(\tilde{r}_t - \bar{r})(\tilde{r}_t - \bar{r})^\top\Big], \qquad \rho_{ij} = \frac{\Sigma_{ij}}{\sqrt{\Sigma_{ii}\,\Sigma_{jj}}}.$$

Ticker	ENEL.MI	NOVN.SW	SAN.MC	SIE.DE	XLB	XLE	XLF	XLI	XLK	XLV	XLY
ENEL.MI	1	0.361204	0.501593	0.458623	0.346312	0.230611	0.358814	0.343259	0.276852	0.290291	0.301715
NOVN.SW	0.361204	1	0.306194	0.354103	0.251305	0.1754	0.284699	0.26837	0.196442	0.385811	0.199246
SAN.MC	0.501593	0.306194	1	0.573796	0.436645	0.391446	0.512861	0.44219	0.287997	0.271441	0.332658
SIE.DE	0.458623	0.354103	0.573796	1	0.451904	0.325824	0.448015	0.462117	0.382849	0.295313	0.391424
XLB	0.346312	0.251305	0.436645	0.451904	1	0.652019	0.786318	0.856423	0.658485	0.640719	0.708362
XLE	0.230611	0.1754	0.391446	0.325824	0.652019	1	0.634183	0.643472	0.414411	0.418234	0.45907
XLF	0.358814	0.284699	0.512861	0.448015	0.786318	0.634183	1	0.852244	0.634363	0.639358	0.71144
XLI	0.343259	0.26837	0.44219	0.462117	0.856423	0.643472	0.852244	1	0.711997	0.670856	0.760626
XLK	0.276852	0.196442	0.287997	0.382849	0.658485	0.414411	0.634363	0.711997	1	0.624415	0.819199
XLV	0.290291	0.385811	0.271441	0.295313	0.640719	0.418234	0.639358	0.670856	0.624415	1	0.616167
XLY	0.301715	0.199246	0.332658	0.391424	0.708362	0.45907	0.71144	0.760626	0.819199	0.616167	1

Figura 6: Matriz de correlaciones de los activos. Fuente: Elaboración propia.

Aquí se observa el bloque común correspondiente a los activos estadounidenses y en la ausencia de un factor geográfico compartido entre las empresas europeas. Este hallazgo resulta relevante al realizar el *backtesting* y validar el modelo, ya que sugiere que la red podría haber aprendido un patrón asociado al factor geográfico, el cual debe considerarse en la interpretación de los resultados.

Esta base de datos procesada alimenta de forma directa el ventaneo y escalado de las entradas de las redes recurrentes, minimizando sesgos de estimación y problemas numéricos en el entrenamiento. Todo el flujo anterior queda automatizado para favorecer la reproducibilidad del proyecto y parametrizado de forma que permita la realización de estudios contrafactuales.

3.2. Construcción de la LSTM de retornos

En esta sección se describe el proceso general de construcción de la red, desarrollándose los detalles de arquitectura, hiperparámetros y problemas de desajuste en el Apéndice B.

3.2.1. Ventaneo y escalado

Una vez se tiene bien definido el *dataframe* (conjunto de datos) donde se reflejan los retornos logarítmicos diarios de los activos, es momento de personalizar las *features* al

propósito actual: predicción.

El primer paso es hacer uso de un **escalador** de datos, para homogeneizar las escalas y magnitudes. Recurrir a este método es esencial por varios motivos:

- **La estabilidad numérica:** puede desvanecerse o explotar para magnitudes de datos muy diversas.
- **Velocidad de convergencia:** el algoritmo de descenso es más rápido con homogeneidad de escalas.
- **Regularización:** evitar que las variables con mayor varianza dominen la función de pérdida.

El escalador usado ha sido el `RobustScaler`, ya que es más robusto tratando *outliers* frente a otras opciones como el `StandardScaler`. Su definición formal y parametrización se detallan en el Apéndice B.

Una vez procesados los datos, se sigue con la segmentación del conjunto. Como se mencionó previamente, la arquitectura de LSTM "mira" una ventana de L días pasados y realiza una predicción para un horizonte h. El primer paso lógico es, pues, la construcción de estas *ventanas móviles* a partir de los datos. En general, para un *dataset* de tamaño T, el número de ventanas viene dado por:

$$windows = T - L - h + 1$$

Por lo tanto, en este caso se contará con 3068 ventanas de datos por activo, cantidad suficiente para el propósito de este trabajo.

Se hace un split *train/validation/test* con las siguientes características:

- **Train (70 %, hasta el 19/01/2022).** Conjunto empleado para el ajuste de los parámetros de la red (entrenamiento propiamente dicho).
- **Validation (15 %, hasta el 30/11/2023).** Conjunto destinado a la selección de hiperparámetros y a la toma de decisiones de diseño (arquitectura, regularización y *early stopping*), sin utilizar información del *test*.
- **Test (15 %, hasta el 12/11/2025).** Conjunto reservado para la evaluación final fuera de muestra. Sus resultados se reportan como evidencia del desempeño del sistema en condiciones no vistas durante el ajuste.

En la figura 7 se puede ver la serie de Banco Santander con los momentos exactos donde se hace la separación de los conjuntos:

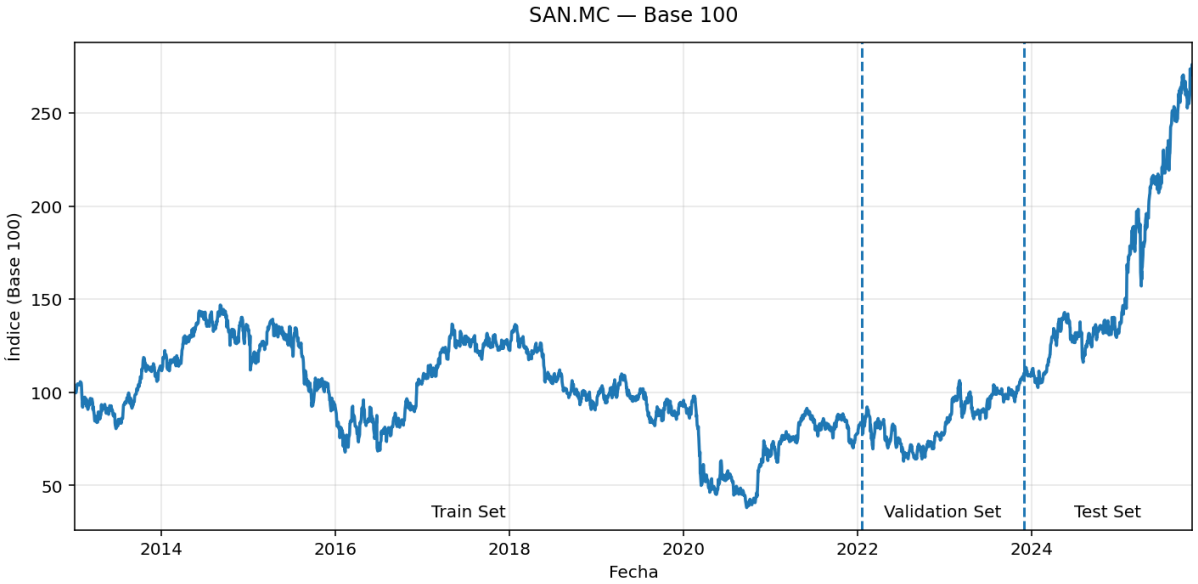

Figura 7: Valor de la acción de Banco Santander en base (01/01/13=100). Fuente: Elaboración propia.

A continuación, se concreta una arquitectura para la LSTM de retornos. Gracias a las librerías de `Keras` y `TensorFlow` este proceso se simplifica. En el cuadro 5 del Apéndice B se detalla la elección de hiperparámetros usados. La configuración inicial sigue la práctica habitual, aunque se pueden modificar hiperparámetros según el desempeño mostrado.

Durante el entrenamiento apareció un problema recurrente: el colapso de las predicciones en un valor constante con sesgo de signo. Para solucionarlo se modificó el escalado y se detrayó un componente de deriva de la variable predictora, logrando entonces una mejora en el desempeño (pasando de un *hit ratio* del 50 % a uno del 70 %). En el Apéndice B se amplía la solución acometida para este problema.

Por último, definimos las métricas de evaluación que se usarán para evaluar la calidad de la red. Estas son el *hit ratio*, el error absoluto medio y la raíz del error cuadrático medio. En el cuadro 7 del Apéndice B se detalla su cálculo.

3.3. Construcción de la LSTM de riesgos

3.3.1. Ventaneo y escalado - La NLL

A continuación se diseña la red de riesgos del modelo a través de la desviación típica, entendida como *proxy* de incertidumbre futura.

Tras recuperar los datos del archivo, y como antes, se aplica un escalado de los datos únicamente a las variables de entrada. Es decir, se escalan exclusivamente las ventanas temporales X, mientras que se mantiene en su escala original la variable objetivo y correspondiente a los retornos realizados y se guarda el escalador para el *backtest*. La motivación es doble (económica y estadística), ya que en esta etapa la magnitud absoluta de los retornos es relevante, pues determina la penalización en la función de pérdida.

En este caso, la variable y consiste en los retornos realizados con horizonte $h = 1$ para cada ventana de tamaño L. La red aprenderá a asignar a cada historial un nivel de **varianza condicional** $\hat{v}$ (y, por extensión, una volatilidad $\hat{\sigma} = \sqrt{\hat{v}}$). Aquí no se busca predecir el retorno, sino su **dispersión esperada**. En la práctica, el valor $\hat{\sigma}$ debe ser grande en periodos de alta volatilidad y pequeño en periodos de calma.

En este caso la función de pérdida empleada en el algoritmo es la *negative log-likelihood (NLL)* gaussiana, cuyo objetivo es evaluar la coherencia entre la varianza condicional predicha por el modelo y los retornos efectivamente observados. Su expresión es:

$$\text{NLL} \;=\; \tfrac{1}{2}\log \hat{v} \;+\; \tfrac{1}{2}\frac{r^2}{\hat{v}}\,.$$

La elección se basa en que la NLL refleja de manera natural la tensión entre ambos componentes del riesgo: si el retorno observado r es grande pero la varianza predicha $\hat{v}$ es pequeña, el término $r^2/\hat{v}$ se dispara y la pérdida aumenta, empujando al modelo a elevar su estimación de riesgo. Si por el contrario, $\hat{v}$ es mucho mayor que lo que muestra la serie, entonces domina el término $\log(\hat{v})$, que penaliza dicha sobreestimación y fuerza a disminuirla. De este modo, la red aprende un equilibrio estadístico adecuado entre ajuste a los datos y calibración del riesgo.

Por último, remarcar que este enfoque es conceptualmente distinto al modelo de retornos previo. Antes la red trataba de predecir la dirección, aquí se trata de estimar la *confianza* o incertidumbre asociada al mismo.

A continuación se crean las ventanas con $L = 240$, ya que un *lookback* más corto (L=60) mostró un desempeño insuficiente para calcular dinámicas relevantes. Con un tamaño de serie $T = 3120$ y $h = 1$, el número de ventanas es:

$$\text{windows} = T - L - h + 1 = 3120 - 240 - 1 + 1 = 2980$$

Pese a ser menor que en la red de retornos, el tamaño muestral es suficiente para estimar la dinámica de volatilidad a $h = 1$. El *split* temporal es igual que el anterior (70 %/15 %/15 %).

En este caso, las métricas de evaluación son el error absoluto medio y la función QLIKE que, en línea con la NLL, relaciona volatilidad predicha y riesgo. En el Apéndice B se detallan la configuración concreta de la red y la formulación matemática de las métricas empleadas.

3.4. Del *Machine Learning* a la optimización de carteras

Una vez explicado cómo funcionan los motores del sistema, se explica cómo traduce el *meta-modelo* estas predicciones para tomar decisiones económicas. El esquema de decisión propuesto, al que se denominará *modelo inspirado en Sharpe*, se describe a continuación.

Se define el **Ratio de Sharpe** como:

$$Sharpe = \frac{r - r_f}{\sigma}$$

donde r es la rentabilidad de un cierto activo, σ es su volatilidad y r_f es la rentabilidad del activo libre de riesgo (por ejemplo los bonos del tesoro). Esta métrica creada por Sharpe (1994) es una excelente medida del rendimiento por unidad de riesgo asumido que da un activo. En otras palabras, indica hasta qué punto el rendimiento de una inversión compensa al inversor por asumir riesgo en su inversión.

En este trabajo se fija $rf = 0$, pero queda parametrizado en el código para análisis alternativos. En la práctica este cambio no altera sustancialmente la comparativa entre estrategias.

Considérese un día cualquiera, antes de la apertura del mercado se deberá determinar la composición de la cartera para dicha jornada (o a horizonte vista h). Para ello, se analizan las ventanas temporales (60 días para retornos y 240 para volatilidades) y se generan predicciones utilizando las dos redes neuronales del modelo. De esta forma, se obtienen las predicciones para un conjunto de activos. Representándolas en forma vectorial, $\hat{\mu}$ son los retornos esperados y $\hat{\sigma}$ la volatilidad diaria (porque se ha escogido $h = 1$). El modelo hace lo siguiente:

$$s_i = \frac{\hat{\mu}_i}{\hat{\sigma}_i + \epsilon}$$

que se denominará la *señal emitida por el activo i*. Nótese que, cuanto mayor es este ratio (inspirado en el de Sharpe), mayor es la señal que emite ese activo. Agregadamente, s es un vector de dimensión N (número de activos considerados).

La señal normaliza retorno por riesgo, haciendo comparables las predicciones entre activos. Para este fin se recurre a una función *softmax con temperatura* como sigue:

$$w_i = \frac{exp(\tau s_i)}{\sum_{j=1}^{N} exp(\tau s_j)}$$

Esta transformación es ventajosa, ya que garantiza positividad y normalidad en los pesos, al tiempo que es una función monótona que no altera el *ranking* de activos. Lo que la diferencia de una transformación lineal "proporcional" es que esta hace una asignación más suave, que evita concentraciones extremas.

Además, τ es un parámetro de la concentración de activos, que se denomina *temperatura*. Cuando $\tau \to 0$ se observa que $w_i \to 1/N$, dicho de otro modo, los pesos tienden a ser iguales. Por el contrario, cuando este parámetro toma valores elevados la concentración hacia un solo activo dominante es mayor, amplificando las diferencias en el ratio precedente.

Económicamente, τ puede interpretarse como un parámetro de concentración en la asignación de pesos del modelo. En otras palabras, τ no se aprende en el entrenamiento, sino que se fija exógenamente según el perfil de riesgo.

La elección de τ responde a una cuestión de preferencias, pues regula la concentración de activos. Cabe esperar que valores más elevados impliquen mayor volatilidad de la curva de capital, aunque la validez estructural del modelo debería mantenerse. En análisis internos ligeros, se ha observado que valores de τ cercanos a 5 o 6 ofrecen un equilibrio

razonable.

Estos elementos componen el núcleo del modelo y explican el mecanismo por el que se traducen predicciones sobre retorno y riesgo en señales económicas y estas, a su vez, en pesos de cartera. Además, después de este proceso se pueden configurar varias opciones extras que enriquecen el modelo financiero: limitación de peso de un mismo activo en la cartera (w_{max}) y del número de activos máximos considerados k, objetivos de volatilidad (anualizada) máxima soportada por la cartera, estrategias *long-short* o *long-only* y opciones de efectivo y apalancamiento.

Estos son parámetros que el usuario especifica en función de su perfil de riesgo, pero no componen la estructura fundamental del sistema, aunque sí lo enriquecen notablemente. En los experimentos base se fija $\tau = 6$, $w_{max} = 0,3$, $k = 4$ y objetivo de volatilidad anual del 15 %. El mecanismo concreto de cada uno se explica en detalle en el Apéndice C.

Sección 4. Resultados

En esta sección se presentan los resultados económicos del sistema completo a través de la prueba de *backtest*, ya que son la validación relevante para el desempeño global del modelo. Las métricas no financieras y los resultados de validación de las LSTM se han desarrollado en el Apéndice D.

Como contexto, la red de retornos alcanza un ratio de acierto cercano al 70 % y un MAE del 0, 8 %. La red de riesgos reporta un MAE en torno al 0, 7 % y supera al *baseline* (media exponencial) en términos de QLIKE. En conjunto, ambas redes capturan señal direccional y de riesgo en los activos de entrenamiento. A partir de aquí, la cuestión relevante es si esa señal se traduce en rentabilidad ajustada al riesgo en cartera, es decir, en el desempeño financiero del meta-modelo global.

4.1. *Backtesting* y validación global

Un **backtest** es un método de evaluación que consiste en aplicar un modelo financiero o una estrategia de inversión a datos históricos con el fin de analizar su desempeño pasado. Esta técnica permite estimar la viabilidad y la rentabilidad, pero también los riesgos que se están asumiendo.

Dada la dimensionalidad del modelo, resulta necesario definir qué escenarios de mercado es interesante considerar y qué opciones y limitaciones se presentan. Globalmente, se van a plantear dos universos; el conocido, con los activos que el modelo ha visto durante el entrenamiento; y el desconocido, donde se trabajará 11 activos diferentes, aunque representan a los sectores y geografías que ya conoce. El objetivo de esta segunda prueba es evaluar la **capacidad de generalización** fuera de la muestra de activos con que se ha entrenado.

Otra prueba relevante es la de comparar la estrategia *long-only* frente a *long-short*, pues esto permite evaluar la capacidad del modelo de predecir tanto al alza como a la baja, evidenciando, o no, su robustez ante diversos regímenes de mercado. Aunque podrían explorarse más configuraciones, el análisis se encuentra limitado por el coste computacional.

Llevar a cabo un examen de este tipo no es sencillo y, en este sentido, Campbell (2005) propone algunos procedimientos y buenas prácticas para realizarlos de forma rigurosa, tales como separar evaluación de rendimiento y riesgo o incluir el semáforo regulatorio. Aunque no se incorporarán todas las recomendaciones del autor, muchas de las métricas señaladas resultan relevantes y se tendrán en cuenta en este trabajo. Así, se exponen los principales indicadores usados:

1. **Métricas visuales.** Aunque no son concluyentes, aportan evidencia preliminar y facilitan la detección de fallos evidentes. Algunas son:
    - **Gráficos de ajuste.** Se trata de reconvertir los retornos predichos por el modelo en precios de activos (en moneda local), y superponerlos con la serie histórica del valor. Se espera que ambas series sean cercanas, de forma que una mayor superposición sugeriría una mejor capacidad predictiva. Para reconstruir

precios a partir de retornos logarítmicos se suman los log-retornos y se exponencian, aplicando además una corrección para evitar desfase por acumulación de error. El procedimiento matemático completo se recoge en el Apéndice E.

- **Curvas de retorno.** Se trata de una de las métricas más intuitivas y muestra qué retorno se obtendría si se eligiera el modelo frente a los *benchmarks* seleccionados (en este caso el S&P500 por ser un índice de referencia global y el IBEX 35 por ser el selectivo español). Si el valor de la cartera se mantiene la mayor parte del tiempo por encima de sus rivales, entonces puede considerarse que la estrategia es satisfactoria.
- **Otras gráficas.** Se presentan también otras gráficas de interés como la comparativa *short vs long* o los *drawdowns* del modelo.

2. **Métricas de rendimiento.** Son las que cuantifican la rentabilidad y eficiencia general de la estrategia. Se incluyen:

   - **CAGR (Compound Annual Growth Rate).** Tasa de crecimiento anual compuesta de la curva de capital.
   - **Volatilidad anualizada.** Desviación típica de los retornos diarios escalada a un año.
   - **Ratio de Sharpe.** Rentabilidad media anualizada dividida por la volatilidad anualizada; mide rentabilidad ajustada al riesgo.
   - **Max Drawdown (MDD).** Mayor caída porcentual desde un máximo histórico de la curva de capital.
   - **Ratio de Calmar** Relación entre CAGR y el valor absoluto del MDD; mide la eficiencia entre retorno y caída máxima.
   - **Hit ratio.** Porcentaje de días en los que el modelo obtiene rentabilidad positiva.
   - **Turnover.** Rotación media de la cartera. Es importante resaltar que en los backtests presentados no se han incorporado explícitamente costes de transacción como comisiones o *slippage*. Esta decisión se adopta para aislar el efecto predictivo del modelo y su lógica de asignación de pesos. No obstante, el *turnover* reportado aproxima la intensidad operativa del sistema y, por tanto, del potencial efecto de dichos costes. Así, estrategias con mayor rotación podrían ver reducida su rentabilidad neta.
   - **Betas de mercado.** Sensibilidad del modelo frente a los índices S&P 500 e IBEX 35, indicadores de exposición direccional.

   Se incluirá el ratio de Sharpe global del modelo, aunque no sea la métrica más informativa en este caso. Esto se debe a que el experimento se hace con un límite a la volatilidad anual del 15 %.

3. **Métricas de gestión de riesgo.** Evalúan la calibración del riesgo extremo y la consistencia del modelo en cola, y muchas se basan en el concepto de *Value at Risk (VaR)*. El *Value at Risk* (VaR) al nivel de confianza $1-\alpha$ y horizonte temporal h es el umbral $\mathrm{VaR}_{\alpha,h}$ tal que la probabilidad de observar una pérdida superior a dicho umbral es α:

$$\mathbb{P}\Big(L > \mathrm{VaR}_{\alpha,h}\Big) = \alpha,$$

donde L denota la pérdida de la cartera.

- **Número de excepciones VaR ($\alpha = 1\,\%$).** Número de días en que la pérdida supera el umbral previsto.
- **Prueba de Kupiec.** Contrasta si la frecuencia observada de violaciones (exceptions) del VaR coincide con el nivel de confianza esperado α.
- **Prueba de independencia (LR_ind).** Verifica si las excepciones se agrupan en el tiempo (clustering).
- **Expected Shortfall (ES).** También conocido como pérdida esperada condicional, complementa al VaR al medir la pérdida media que se produce en los peores casos, es decir, cuando las pérdidas superan el umbral del VaR.
- **Semáforo de Basilea.** Indicador regulatorio visual (verde, amarillo o rojo) según el número de violaciones recientes.

4.1.1. Resultados con activos conocidos *long-only*

En la figura 8 se recoge el ajuste de las predicciones de la red de retornos superpuestas con la serie original en moneda local para cada uno de los activos.

Figura 8: Ajuste general de las predicciones en long-only y activos conocidos. Fuente: Elaboración propia.

La figura sugiere un ajuste razonable. Como se observa, en todos los activos la red se superpone bien, lo que es consistente con una buena adaptación a la serie. En algunos activos y momentos puntuales, como ENEL o Novo Nordisk en periodos de caída abrupta (2020 por ejemplo), se observa que la red ha tendido a subestimar la intensidad de la caída. Como resultado se obtiene una red ajustada que, aún con margen de mejora, podría usarse para predicción.

Esa primera gráfica era un indicador exclusivo de la calidad de predicción, no del modelo de inversión. En la figura 9 se ve la comparativa entre evolución de la cartera y el *S&P500* y el *IBEX35*, los dos *benchmarks* que enfrenta el modelo.

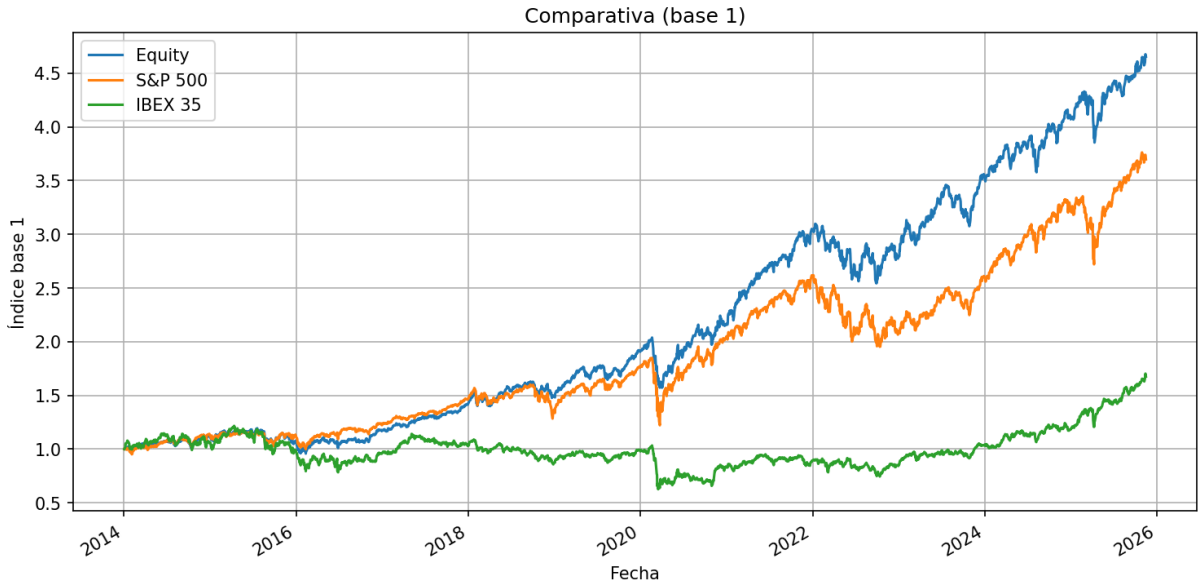

Figura 9: Evolución de la cartera long-only en relación a los índices S&P 500 e IBEX 35. Fuente: Elaboración propia.

La evidencia visual sugiere la calidad del meta-modelo global. Primeramente, la cartera supera con holgura en términos globales a los índices de referencia, alcanzando aproximadamente 4,5 veces su valor inicial en el transcurso de casi doce años. Segundo, entre 2014 y 2019 aproximadamente la cartera va en línea con el *S&P500*, lo cual indica que era el selectivo más elegido por el modelo, seguramente por el mayor impulso y atractivo de las empresas que lo componen (el modelo apostaba más por los ETF sectoriales americanos). Sin embargo, a partir de 2020 la situación cambia y la cartera empieza a despegarse de Estados Unidos. Se observa un patrón particularmente llamativo: la cartera evoluciona en consonancia con el índice, pero las caídas son menos pronunciadas y las ganancias se intensifican. Una interpretación plausible es que, en el periodo post-pandemia el mecanismo de riesgos desplaza pesos hacia activos relativamente más defensivos o reduce su exposición.

Como resultado, se tiene una curva que supera a lo que hubiera representado una estrategia pasiva de inversión, es decir, invertir en un selectivo y dejarlo crecer (**buy and hold**). La lectura extraída es que el sistema es capaz de identificar momentos de alta volatilidad y, principalmente, logra seleccionar alternativas que resultan ventajosas en dichos contextos de mercado.

La figura 10 muestra la volatilidad (anualizada) asumida por el modelo, dado el límite en el 15 % (en otras palabras, ninguna de sus carteras diarias suponen una volatilidad conjunta superior a este valor):

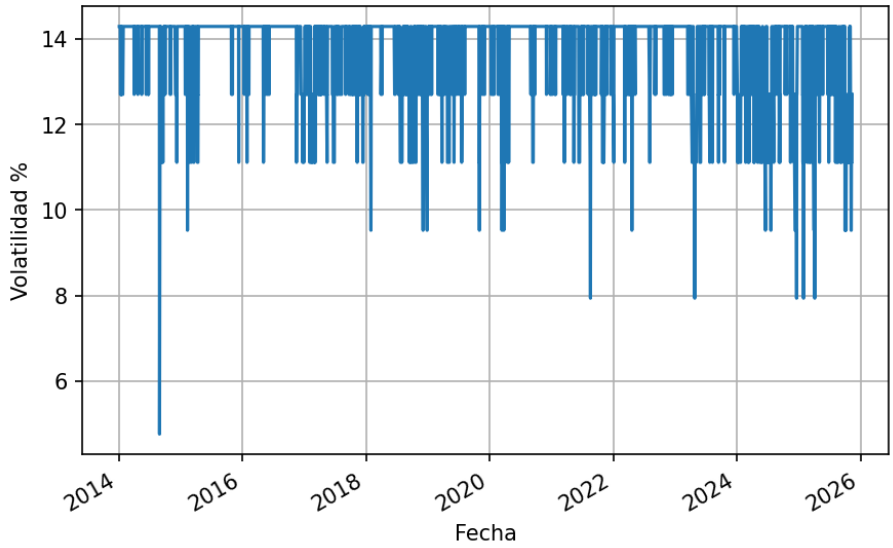

Figura 10: Volatilidad diaria (anualizada) asumida por el modelo. Fuente: Elaboración propia.

El resultado es el esperado, en tanto que el límite de riesgo que se puso se está cumpliendo. Se compara con la volatilidad realizada del mercado, que se ve en la figura 11.

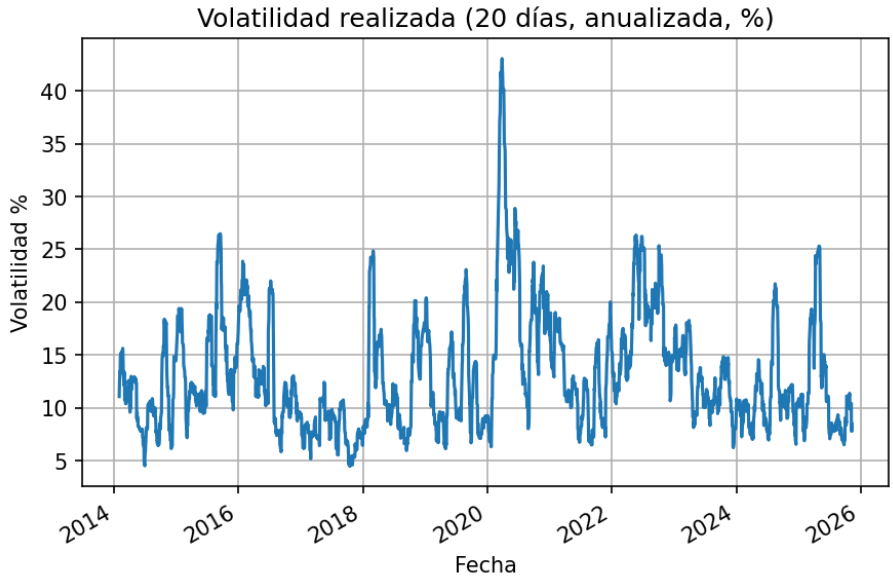

Figura 11: Volatilidad (anualizada) del mercado a partir de 2014. Fuente: Elaboración propia.

Se observa que los momentos más convulsos del mercado son aplacados por el modelo, y que en los tiempos de menor riesgo (como finales de 2014), se ajusta bien a las observaciones reales. Esto va en línea con lo ya señalado, y permite al inversor cubrirse en cierto modo de periodos más arriesgados en el mercado.

Por último, se estudia la gráfica de *drawdowns* del modelo. Un ***drawdown*** es la caída porcentual sufrida desde un máximo previo e indica cuánto ha descendido la cartera desde el último pico antes de volver a recuperar, y viene dado por:

$$\text{Drawdown}_t = \frac{\text{Equity}_t - \text{máx}_{s \leq t}\, \text{Equity}_s}{\text{máx}_{s \leq t}\, \text{Equity}_s}$$

Cuando las ganancias son máximas, este indicador valdrá 0 %, mientras que en momentos de bajada será negativo. En la figura 12 se muestran los *drawdowns* sufridos por el modelo.

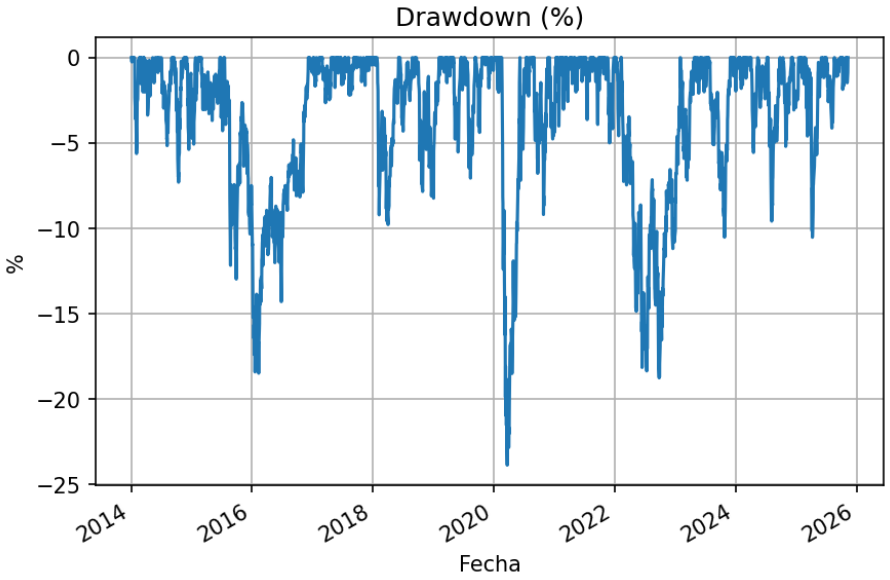

Figura 12: Drawdowns reportados por el modelo long-only. Fuente: Elaboración propia.

Se observa que, en condiciones normales, las correcciones sufridas se mantienen en rangos moderados, que oscilan entre un −5 % y un −10 %. En episodios más convulsos (como en 2016, la crisis del COVID-19 en 2020 o la volatilidad macro de 2022) aparecen retrocesos más acusados, alcanzando un máximo cercano al 23 %. Este valor corresponde al **máximo drawdown**. En general, y pese a estas perturbaciones, el modelo muestra una capacidad de recuperación relativamente rápida tras cada episodio, lo que es indicio de una estrategia estable y resiliente en distintos entornos de mercado.

En el cuadro 1 se presentan las métricas de rendimiento del modelo.

CAGR	Vol_ann	Sharpe	MaxDD	Calmar	Hit_ratio
0,143086	0,138263	1,033703	-0,22921	0,624255	0,556323

Turnover	Gross_mean	Net_mean	Beta_SP500	Beta_IBEX35
0,399512	0,898592	0,898592	0,640181	0,403709

Cuadro 1: Métricas de rendimiento en el escenario *long-only* con activos conocidos. Fuente: Elaboración Propia.

Las métricas son coherentes con lo observado en las figuras: CAGR en torno al 14 % con una volatilidad anualizada cercana al 14 %, que concuerda con el objetivo fijado de volatilidad. El ratio de Sharpe concuerda con lo esperado. Más interesante es el ratio de Calmar, en torno a 0,62, que relaciona este crecimiento con un máximo *drawdown* del −23 % y refleja una estrategia rentable, pero que sufre en episodios de estrés.

El *hit ratio ronda* el 56 %, es decir, el modelo gana algo más de la mitad de los días, lo cual encaja con la idea de una estrategia que no busca acertar siempre, sino funcionar bien en los momentos más regulares del mercado. La rotación media de la cartera (turnover = 0.40) indica un nivel de operativa moderado evitando una estrategia excesivamente activa y la exposición bruta y neta en torno al 0,9 muestra que, en ausencia de cortos, la cartera está casi siempre invertida, con betas frente al $S\&P500$ y al $IBEX35$ inferiores a 1, lo que sugiere una exposición direccional relevante pero algo más contenida que la de los índices de referencia.

Nº de excepciones	Kupiec	Independencia	Expected Shortfall	Semáforo
173	337.81 (p<1e-16)	41.16 (p=1.4e-6)	0.0203	Rojo

Cuadro 2: Análisis del *Value at Risk* con posiciones largas y activos conocidos. Fuente: Elaboración Propia.

Los resultados del *Value at Risk* mostrados en el cuadro 2 ponen números a lo que ya se había anticipado: **el sistema opera bien en entornos de mercado regulares, pero sufre ante caídas repentinas.** Se da un número de excepciones de 173, lo que representa aproximadamente un 6 % de las observaciones. La prueba de Kupiec indica que la frecuencia de excepciones observadas difiere significativamente de la esperada, mientras que la prueba de independencia sugiere cierta dependencia temporal entre eventos extremos (p = 1,4e-6), lo que podría reflejar *clustering* de volatilidad. El *Expected Shortfall* (ES) resultó en 0,0203, proporcionando una estimación de la pérdida promedio condicional en los peores casos. Finalmente, el semáforo de riesgo asignado es "Rojo", indicando que, aunque el modelo es estable en condiciones normales, se deben considerar ajustes adicionales para escenarios de riesgo extremo.

Esto constituye una limitación relevante, ya que aunque el retorno es considerable, se está exponiendo mucho ante una situación adversa imprevista. Pero, en cualquier caso, abre una vía de mejora: cómo fortalecer el modelo en situaciones de *shock* sin sacrificar capacidad predictiva en condiciones de regularidad.

4.1.2. Resultados con activos conocidos *long-short*

En esta variante cambian las reglas financieras al permitirse cortos, manteniéndose constantes las predicciones de las LSTM. Es decir, no cambia nada relacionado con el funcionamiento de las LSTM, por lo que la calidad de predicción va a ser la misma. No se mostrarán entonces las gráficas de ajuste porque son iguales, pero sí que se verá la comparativa con los *benchmarks* (figura 13).

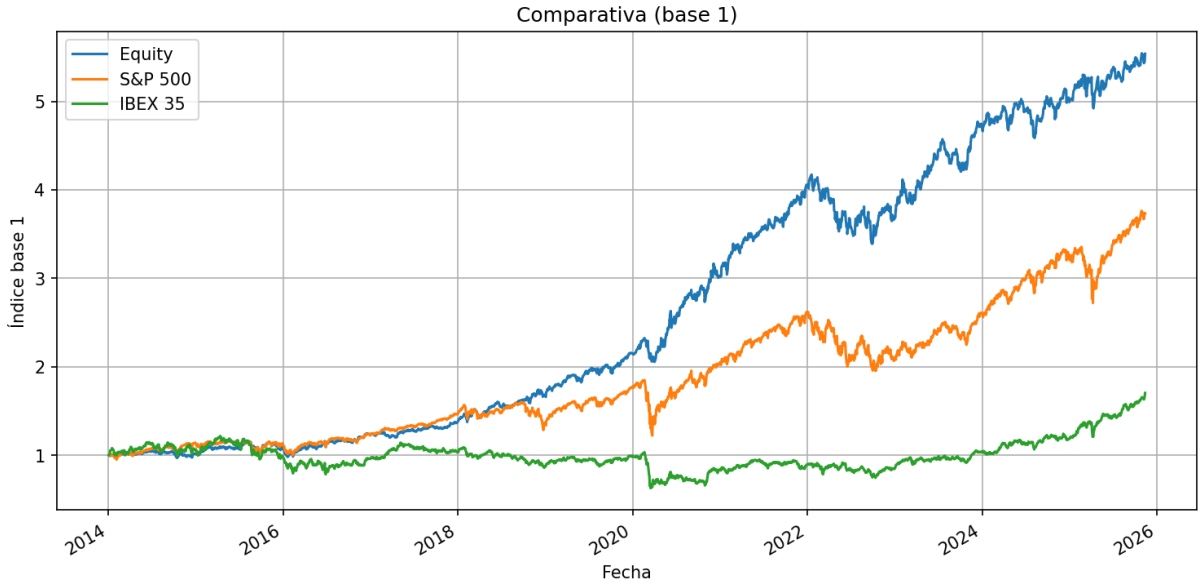

Figura 13: Evolución de la cartera con long-short y activos conocidos. Fuente: Elaboración propia.

En primer lugar, el modelo bate nuevamente a los *benchmarks* seleccionados y esta vez con algo más de margen que el anterior. También se observa que las caídas en casos de crisis no son tan pronunciadas como antes, lo que es una señal de que el modelo, aunque poco reactivo ante grandes caídas, sabe identificar un mercado bajista y ponerse en corto. Véase por ejemplo la corrección sufrida en 2025, que fue menos pronunciada. En la figura 14 se recoge la comparativa directa con la estrategia precedente.

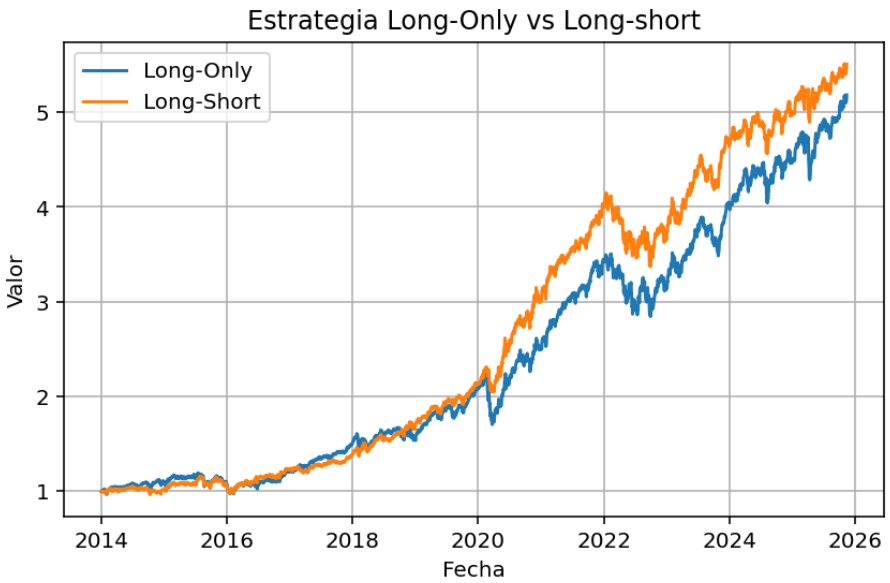

Figura 14: Resultados de la estrategia long-only contra long-short. Fuente: Elaboración propia.

Con la estrategia de cortos se observan caídas menos abruptas y, en general, un mejor desempeño. No obstante, hay un momento específico después del periodo bajista de 2016 donde la estrategia en corto lo hace peor que la de solo en largo. Esto podría deberse a que el modelo no supo reaccionar rápido al "cambio de rumbo" del mercado. Concretamente,

al igual que el modelo no está entrenado para reaccionar rápido frente a grandes shocks, podría ser que fuera igual de reactivo frente a cambios de pesimismo general hacia mercados más alcistas. Aunque este comportamiento no se repite en los shocks posteriores, resulta una hipótesis plausible que podría explorarse en investigaciones futuras y servir de base para posibles mejoras.

Por último, se muestra la figura 15 sobre *drawdowns*, ya que para comparar estrategias resulta muy reveladora:

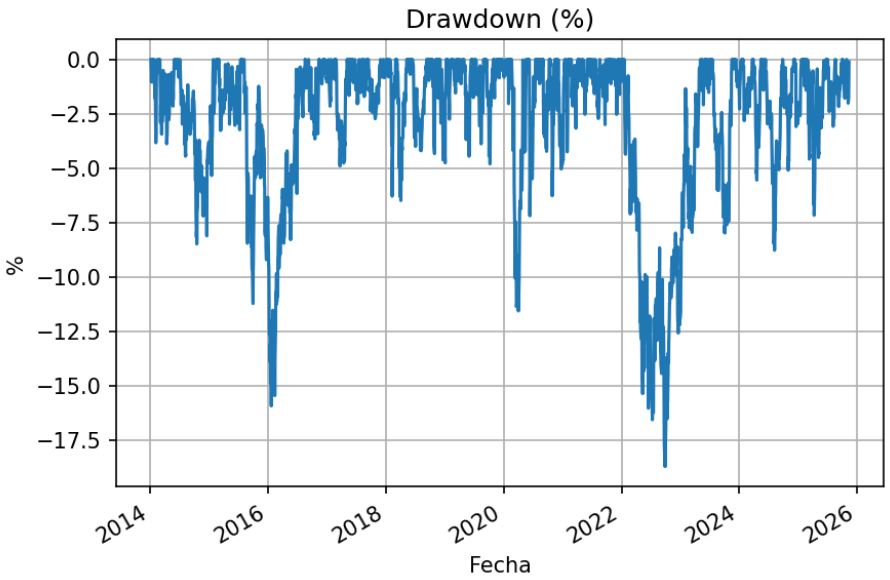

Figura 15: Drawdowns reportados por el modelo long-short con mundo conocido. Fuente: Elaboración propia.

La gráfica revela que la adopción de posiciones en corto ha mejorado la capacidad reactiva del modelo, pudiendo posicionarse mejor cuando detecta un periodo bajista (mejora muy significativa en el momento del COVID). Esto se observa en dos puntos: un menor ***drawdown* máximo** (antes del 23 % y ahora del entorno del 18 %; y una mayor recuperación post-shock que se aprecia en valles más estrechos y puntiagudos).

CAGR	Vol_ann	Sharpe	MaxDD	Calmar	Hit_ratio
0,148287	0,124925	1,166267	-0,17958	0,825749	0,542502

Turnover	Gross_mean	Net_mean	Beta_SP500	Beta_IBEX35
0,506051	0,919375	0,660285	0,446762	0,235986

Cuadro 3: Métricas de rendimiento en el escenario *long-short* con activos conocidos. Fuente: Elaboración Propia.

Al permitir posiciones en corto, y como ya se ha señalado, los resultados permiten apreciar una mejora en la eficiencia. En el cuadro 3 vemos que el CAGR aumenta ligeramente hasta el $14,8\,\%$ y la volatilidad anualizada baja al entorno del $12,5\,\%$, de modo que el ratio de Sharpe sube hasta aproximadamente 1,17 y el ratio de Calmar alcanza

valores cercanos a 0,83. Dicho de otro modo, con una caída máxima más contenida (MDD = -18 %) el modelo consigue generar un crecimiento algo mayor manteniendo simultáneamente pérdidas intermedias menores, alineado con todo lo que se venía anticipando con los gráficos.

El *hit ratio* desciende levemente hasta el 54 %, pero este pequeño sacrificio en porcentaje de días ganadores se ve compensado por una mejor gestión de los momentos convulsos y por una mejor relación rentabilidad–riesgo global. Como cabía anticipar, el *turnover* aumenta (aprox 0,50), reflejando una mayor actividad en cartera asociada a la apertura y cierre de posiciones cortas, aunque este incremento podría implicar un mayor coste operativo debido a comisiones, afectando la rentabilidad neta de la estrategia. Finalmente, la exposición bruta se mantiene en niveles similares, pero la exposición neta baja claramente (en torno a 0,66) y las betas frente a $S\&P500$ e $IBEX35$ caen por debajo de las de la versión *long-only*, lo que indica una estrategia menos direccional y con un componente más marcado de apuestas relativas entre activos.

Nº de excepciones	Kupiec	Independencia	Expected Shortfall	Semáforo
143	233.32 (p<1e-16)	20.5086 (p=5.94e-6)	0.018	Rojo

Cuadro 4: Análisis del *Value at Risk* con posiciones largas y cortas y activos conocidos. Fuente: Elaboración Propia.

Los resultados mostrados en el cuadro 4 son coherentes con todo lo expuesto previamente: el modelo sigue siendo débil en momentos de ruptura del mercado, pero está suavizando las pérdidas frente al modelo de solo largos. Esto se observa en un menor número de excepciones y, sobre todo, en una pérdida esperada algo menor (en torno al 1,8 %). Se puede concluir, por tanto, que el modelo presenta limitaciones frente a *shocks* extremos, lo que sugiere que, en futuras revisiones o mejoras, esta área debería ser prioritaria.

4.1.3. Resultados con activos desconocidos *long-short*

A continuación, se procederá a evaluar la capacidad de generalización del modelo, examinando si puede trasladar su desempeño predictivo a activos distintos dentro de la misma industria. Para hacer este experimento se han elegido once activos de las mismas geografías (EEUU frente Europa) y de los mismos sectores de actividad. El selectivo resultante es Nvidia, JPMorgan, Walmart, Chevron, Merck Sharp y Caterpillar en EEUU, y BBVA, British Petroleum, Airbus y Sanofi en Europa. Todos son sectores que el modelo conoce. En la figura 16 se muestran los ajustes predictivos.

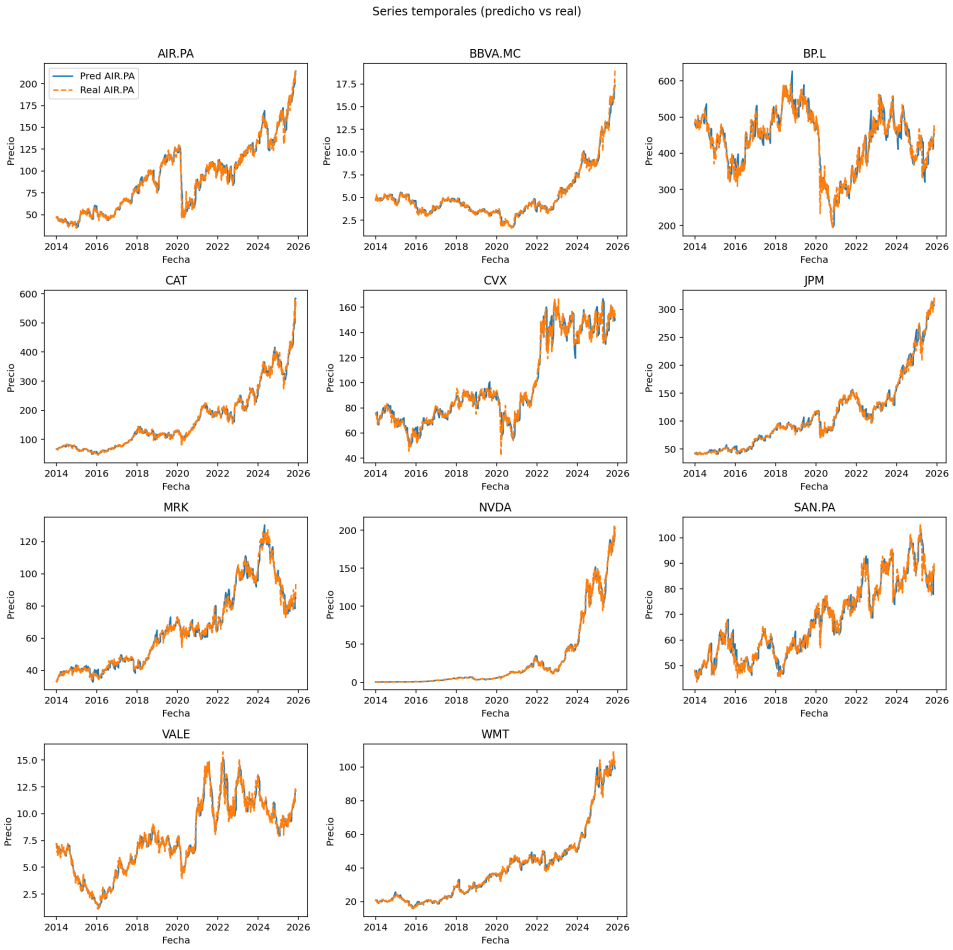

Figura 16: Ajuste general de las predicciones en long-short y mundo desconocido. Fuente: Elaboración propia.

El ajuste es razonable en promedio, pero con fallos relevantes. Se ve en los casos de Chevron (CVX), British Petroleum (BP.L) o Sanofi (SAN.PA), donde hay numerosos picos azules indicando que la red parece no adaptarse a los datos reales en muchos momentos.

En conclusión, la red no exhibe una capacidad de generalización robusta a nuevos activos. En otras palabras, si el modelo es poco reactivo a shocks, también puede serlo ante giros bruscos de régimen.

Para completar la evaluación, se ofrece el resultado del *backtest* financiero frente a los selectivos americano y español (figura 17).

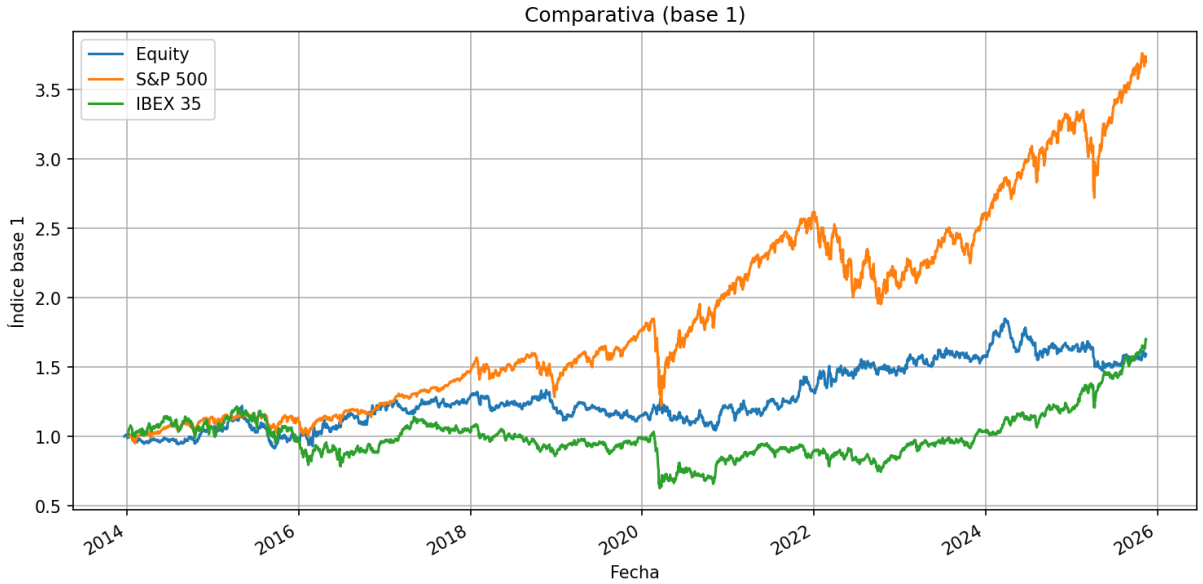

Figura 17: Evolución de la cartera long-short con mundo desconocido. Fuente: Elaboración propia.

El modelo ha mostrado un desempeño inferior frente a los dos índices de referencia utilizados. Por ende, aunque su capacidad predictiva es considerable, la generalización no es estable entre activos, incluso cuando comparten industria y ciertas características.

En conclusión, las LSTM propuestas en este trabajo no generalizan de forma robusta. Esto se debe, en parte, a la configuración elegida y al número de activos. Con un conjunto de datos reducido como el que se usó es normal que ocurra esto, aunque no invalida la utilidad del enfoque dentro del universo de entrenamiento.

Como experimento complementario, se realizó un *backtest* con apalancamiento y un pequeño análisis de costes. Los resultados del mismo se adjuntan en el Apéndice F.

Sección 5. Limitaciones y líneas de mejora

5.1. Limitaciones

Las limitaciones que este modelo enfrenta son comunes en los modelos de redes neuronales, y más concretamente de las redes recurrentes.

1. **El coste computacional.** Entrenar estos modelos exige recursos y tiempo significativos, y es algo de lo que no siempre se dispone. En particular, el ajuste de hiperparámetros es el componente más costoso. En el cuadro 13 del Apéndice Gse recogen todas estas variables que presenta el meta-modelo financiero completo (incluyendo tau, k, w_{max} y ventanas). En los escenarios comparados se modifican únicamente los activos considerados, las posiciones en largo/corto y el apalancamiento y el resto de parámetros permanece por defecto.

   Considérese que, si por cada parámetro se plantearan 3 o 4 posibles configuraciones se darían

   $$total = 3^{39} = 4{,}0525552e + 18$$

   iteraciones del modelo, algo del todo inviable. Existen alternativas, como la búsqueda bayesiana de hiperparámetros, aunque el coste sigue siendo elevado.

2. **La escala de la muestra.** Una muestra de 11 activos, aunque haya diversidad de geografías y de sectores, no representa fielmente el mercado real. La escalabilidad que se necesitaría para lograr un modelo que opere en un mercado verdaderamente abierto exigiría recursos computacionales muy superiores a los disponibles en este trabajo. No obstante, existe la posibilidad de escalar el proyecto con equipos más potentes.

   Cabe señalar además una conclusión fundamental extraída, y es que se debe usar una red entrenada concretamente para los activos que se quiera predecir. En este planteamiento, el rendimiento cae al trasladar el modelo a activos no vistos sin un entrenamiento específico.

3. **El problema del ruido.** Las series temporales financieras son sistemas complejos, donde interfieren muchas variables que dificultan su predicción. El preprocesado prioriza robustez frente a atípicos; a cambio, reduce la efectividad en episodios extremos. Conseguir un modelo equilibrado en ambas cuestiones supondría recurrir a arquitecturas y procesos más complejos, que no son abarcables por este trabajo.

4. **La interpretabilidad del modelo.** Una de las limitaciones inherentes a las redes neuronales, y especialmente a las LSTM, es su falta de interpretabilidad. A diferencia de los modelos econométricos o de *scoring* tradicionales, donde cada parámetro tiene un significado claro, en una red neuronal el conocimiento se distribuye entre miles de pesos, lo que dificulta extraer conclusiones económicas directas. Existen técnicas de interpretación, como el análisis de sensibilidad o la descomposición de importancia de características, pero ninguna ofrece aún una comprensión completa del proceso de decisión del modelo.

5. **La estabilidad temporal.** Otra limitación relevante es la pérdida de validez del modelo con el paso del tiempo. Los patrones financieros no son estacionarios: los regímenes de mercado, las correlaciones y los comportamientos de los agentes cambian, por lo que un modelo entrenado sobre un periodo histórico puede degradarse si las condiciones estructurales del mercado se modifican. Esto obliga a reentrenar el modelo con cierta frecuencia, con el consiguiente coste computacional y el constante **riesgo de sobreajuste** a nuevos datos.

6. **Costes de transacción y fricciones de mercado.** Los resultados del backtest se presentan en términos brutos (sin comisiones, *spread* ni *slippage*). Aunque en brokers minoristas existen tarifas fijas muy bajas, el coste relevante en términos económicos no siempre desaparece con el tamaño invertido, ya que el *spread* y el *impacto de mercado* suelen escalar con el nominal y con la liquidez del activo. En consecuencia, los resultados deben interpretarse como una cota superior del rendimiento neto, siendo esperable un deterioro mayor cuanto mayor sea la rotación de cartera.

Vistas algunas de las principales limitaciones que presenta esta propuesta y, en general, los modelos de tipo LSTM en series financieras, se presentan algunas de las posibles líneas de mejora que podrían explorarse para tratar de mejorar el rendimiento.

5.2. Líneas de mejora

A pesar de las limitaciones comentadas, el modelo desarrollado demuestra un potencial significativo. Las redes LSTM ofrecen una base sólida para abordar la predicción de variables financieras, si bien su desempeño y aplicabilidad podrían incrementarse mediante distintas mejoras, tanto técnicas como conceptuales. En esta sección se presentan algunas líneas de desarrollo que podrían explorarse en futuras versiones del modelo.

1. **Incorporación de variables exógenas que contextualicen los datos.** Esta es, probablemente, la línea de investigación más prometedora en el ámbito de las LSTM aplicadas a finanzas. Consiste en enriquecer el conjunto de datos con variables de naturaleza económica o financiera que ayuden al modelo a entender el entorno en el que se generan los retornos. Indicadores macroeconómicos (PIB, inflación, tipos de interés), variables de mercado (índices sectoriales, tipos de cambio) o incluso factores técnicos derivados de otros modelos pueden aportar contexto y mejorar la capacidad predictiva de la red.

   Por otro lado, una vía de investigación emergente consiste en combinar datos cuantitativos con información textual, incorporando al modelo variables derivadas del análisis de noticias o de indicadores de sentimiento. Este enfoque se apoya en modelos de *Natural Language Processing* (NLP), que podrían permitir a la red interpretar el tono de las noticias económicas y anticipar movimientos del mercado. Sin embargo, esta integración plantea desafíos computacionales y conceptuales que exceden el alcance de este trabajo.

2. **Modelos híbridos LSTM–GARCH y arquitecturas avanzadas.** Una línea de investigación muy interesante es la de los modelos híbridos, que buscan combinar las ventajas de los enfoques clásicos con la flexibilidad de las redes neuronales. En

particular, las arquitecturas LSTM–GARCH permiten que la red capture dependencias no lineales y de largo plazo, mientras que el componente GARCH proporciona una estimación explícita de la volatilidad condicional. Zhou et al. (2021) proponen un modelo de este tipo, en el que las salidas de un GARCH se incorporan como variables de entrada a una LSTM para predecir la volatilidad futura. Los autores demuestran que este enfoque híbrido mejora significativamente el desempeño respecto a redes puras o modelos econométricos aislados. Dado el interés que suscita esta propuesta, se trató de replicar en este trabajo, aunque sin éxito.

Asimismo, podrían explorarse arquitecturas más complejas, como las *Bidirectional LSTM* (BiLSTM) Schuster and Paliwal (1997), que procesan la información tanto hacia adelante como hacia atrás en el tiempo, o las redes *Attention-based*, que ponderan la relevancia de cada observación dentro de la secuencia. Estas estructuras ofrecen mejoras en precisión y capacidad de representación, especialmente en entornos financieros con alta interdependencia temporal.

3. **Métodos de interpretación e inteligencia explicable.** En un contexto financiero, donde las decisiones deben ser justificables, el hecho de que las redes neuronales operen muchas veces como "cajas negras" es cuestionable. En este sentido, algunos autores como Arsenault et al. (2025) han desarrollado recientemente técnicas de *Explainable Artificial Intelligence* (XAI), como los métodos SHAP, LIME o los mapas de atención, que podrían adaptarse para analizar la sensibilidad del modelo a diferentes entradas. Incorporar estos enfoques ayudaría a interpretar las decisiones de la red y a dotar al sistema de una mayor transparencia analítica.

Sección 6. Conclusiones

A lo largo de este trabajo se ha abordado una pregunta sencilla de formular, pero compleja de responder empíricamente: **¿puede un modelo basado en redes neuronales seleccionar y gestionar una cartera de valores de forma eficiente?** El recorrido ha sido amplio y se ha abarcado desde la teoría financiera clásica, hasta técnicas modernas de aprendizaje profundo y una implementación práctica que ofrece resultados satisfactorios en los escenarios conocidos.

En primer lugar, la construcción de las dos redes LSTM ha permitido capturar tanto la dirección del mercado como su volatilidad. Este doble enfoque ha demostrado ser mucho más rico que utilizar una sola red o depender exclusivamente de modelos más clásicos. Las LSTM han mostrado una buena capacidad para aprender patrones temporales y relaciones entre activos, siempre dentro de las limitaciones razonables del universo reducido con el que se ha trabajado.

En segundo lugar, el meta-modelo financiero inspirado en el Ratio de Sharpe ha resultado eficiente transformando predicciones puramente numéricas en decisiones económicas: seleccionar activos, asignar pesos, gestionar el riesgo y adaptar la exposición según las condiciones del mercado. En conjunto, el sistema ha logrado comportarse de manera coherente, estable y competitiva frente a los *benchmarks* seleccionados, especialmente en el escenario *long-short*.

En tercer lugar, el *backtesting* ha revelado las dos caras del modelo. Por un lado, ha mostrado una estrategia con buen rendimiento, *drawdowns* razonablemente contenidos y una capacidad apreciable para adaptarse a entornos volátiles. Por otro lado, ha puesto de manifiesto limitaciones como debilidad frente a *shocks* bruscos o problemas de generalización a activos nunca vistos.

Este trabajo muestra que, dentro del universo de entrenamiento, las **redes neuronales recurrentes pueden integrarse de manera natural en el diseño de carteras**, aportando flexibilidad y potencia predictiva allí donde los modelos clásicos son más rígidos. Las posibles líneas de mejora, como los modelos híbridos tipo LSTM-GARCH, marcan un camino prometedor sobre un modelo básico satisfactorio.

El modelo presentado debe entenderse como un punto de partida: los resultados indican que la combinación de aprendizaje profundo y teoría financiera constituye un terreno fértil para seguir investigando. Con más datos y capacidad de cómputo, sería plausible desarrollar sistemas capaces de anticipar mejor los regímenes de mercado y gestionar carteras. En este sentido, el trabajo ofrece una base sólida y reproducible para asignar carteras a partir de señales aprendidas, con resultados prometedores y un margen de mejora claramente identificado.

Referencias

Arsenault, P.-D., Wang, S., and Patenaude, J.-M. (2025). A survey of explainable artificial intelligence (xai) in financial time series forecasting. *ACM Computing Surveys*, 57(10):Article 265.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, New York.

Campbell, S. D. (2005). A review of backtesting and backtesting procedures. Finance and Economics Discussion Series 2005-21, Board of Governors of the Federal Reserve System, Washington, D.C. FEDS Working Paper.

Casas, M. C. and Cepeda, E. (2008). Modelos arch, garch y egarch: Aplicaciones a series financieras. *Cuadernos de Economía*, 27(48):287–319.

Chen, K., Zhou, Y., and Dai, F. (2015). A lstm-based method for stock returns prediction: A case study of china stock market. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 2823–2824. IEEE. Shanghai Jiaotong University and MD Anderson Cancer Center.

Chrislb (2025). Multilayerneuralnetworkbigger_english.png. https://es.wikipedia.org/wiki/Archivo:MultiLayerNeuralNetworkBigger_english.png. Imagen publicada el 14/07/2025. Licencia GFDL. Último acceso: 21/11/2025.

Dash, S., Chakravarty, S., Giri, N. C., Agyekum, E. B., and AboRas, K. M. (2024). Minimum noise fraction and long short-term memory model for hyperspectral imaging. *International Journal of Computational Intelligence Systems*, 17(16).

Fischer, T. and Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1992). Multilayer feedforward networks with a non-polynomial activation function can approximate any function. Information Systems Working Paper IS-92-13, Stern School of Business, New York University. Faculty Digital Archive, NYU Libraries.

Markowitz, H. (1952). Portfolio selection. *The Journal of Finance*, 7(1):77–91.

Nelson, D. B. (1991). Conditional heteroskedasticity in asset returns: A new approach. *Econometrica*, 59(2):347–370.

Prater, R., Hanne, T., and Dornberger, R. (2024). Generalized performance of lstm in time-series forecasting. *Applied Artificial Intelligence*, 38(1):2377510. Article e2377510. Published online: 15 Jul 2024.

Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

scikit-learn developers (2023). sklearn.preprocessing.robustscaler. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html.

Sharpe, W. F. (1994). The sharpe ratio. *The Journal of Portfolio Management*. Available at https://web.stanford.edu/~wfsharpe/art/sr/sr.htm.

Yu, Y., Si, X., Hu, C., and Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. *Neural Computation*, 31(7):1–36.

Zhang, Z. and Zohren, S. (2025). *Deep Learning in Quantitative Trading*. Elements in Quantitative Finance. Cambridge University Press.

Zhou, G., Zhang, J., and Liu, Y. (2021). Lstm with multiple garch-type models for financial volatility forecasting. *Expert Systems with Applications*, 170:114520.

Apéndices

Apéndice A. Redes Neuronales Artificiales (RNA)

A.1. Intuición geométrica

Recordemos que en una red "feed-forward" se concatenan una serie de capas compuestas por neuronas que hacen transformaciones sobre las entradas. Cada capa de la red lleva a cabo, en esencia, dos transformaciones sobre los datos. Primero, una transformación lineal, que rota y escala el espacio de entrada; después, una transformación no lineal, que "dobla" el espacio, permitiendo que la red se adapte a relaciones intrincadas.

Al combinar muchas de estas transformaciones, la red es capaz de aproximar funciones de prácticamente cualquier forma, lo que se formaliza en el llamado *teorema del aproximador universal* Leshno et al. (1992). Este teorema establece que una red neuronal con una única capa oculta y suficientes neuronas puede aproximar cualquier función continua definida en un conjunto compacto.

A.2. Entrenamiento y función de error

El aprendizaje de la red se basa en la minimización de una función de error que cuantifica la diferencia entre las predicciones de la red y los valores reales. Una de las más habituales es el error cuadrático medio:

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \left(y_k(\mathbf{x}_n, \mathbf{w}) - t_{nk} \right)^2 ,$$

donde t_{nk} representa el valor objetivo para el patrón n. El proceso de aprendizaje consiste en encontrar el conjunto de pesos $\mathbf{w}$ que minimiza esta función, lo cual se logra habitualmente mediante métodos de descenso del gradiente:

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \, \nabla_{\mathbf{w}} E(\mathbf{w}^{(\tau)}),$$

siendo η la tasa de aprendizaje, que controla la magnitud del ajuste en cada iteración.

A.3. Retropropagación del error

El cálculo eficiente del gradiente se realiza mediante el algoritmo de *retropropagación del error* (*backpropagation*), que aplica la regla de la cadena para propagar los errores desde la salida hacia las capas anteriores. Para cada muestra, se define el error local en la salida como

$$\delta_k = y_k - t_k,$$

y se propaga hacia atrás a las neuronas ocultas:

$$\delta_j = h'(a_j) \sum_{k=1}^{K} w_{kj}^{(2)} \delta_k.$$

Con ello, los gradientes de los pesos se calculan fácilmente como

$$\frac{\partial E_n}{\partial w_{ji}^{(1)}} = \delta_j x_i, \quad \frac{\partial E_n}{\partial w_{kj}^{(2)}} = \delta_k z_j.$$

Este procedimiento permite actualizar todos los pesos de la red de manera eficiente, incluso cuando el número de parámetros es muy elevado.

A.4. Interpretación y propiedades

Entre las propiedades más destacadas de las redes neuronales se encuentran su capacidad para modelar relaciones altamente no lineales, su universalidad como aproximadores de funciones y su entrenamiento eficiente mediante el gradiente descendente. No obstante, estas ventajas van acompañadas de ciertos desafíos, entre los que destaca el riesgo de sobreajuste, lo que hace imprescindible la aplicación de técnicas de regularización adecuadas para garantizar que la red capture patrones estructurales relevantes y no ruido presente en los datos.

En suma, las redes neuronales constituyen un modelo paramétrico no lineal capaz de aprender representaciones jerárquicas de los datos, combinando transformaciones lineales y no lineales cuyos parámetros se ajustan iterativamente para minimizar una función de error. Su flexibilidad y capacidad de generalización las convierten en una herramienta de enorme valor dentro de la inteligencia artificial aplicada a las finanzas.

A.5. La célula LSTM: ecuaciones

Como explica Yu et al. (2019), las ecuaciones que componen la célula LSTM se basan en:

$$\begin{aligned}
f_t &= \sigma(W_f h_{t-1} + U_f x_t + b_f), \\
i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i), \\
\tilde{c}_t &= \tanh(W_c h_{t-1} + U_c x_t + b_c), \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \\
o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o), \\
h_t &= o_t \odot \tanh(c_t),
\end{aligned}$$

donde $\odot$ denota el producto elemento a elemento.

Se podría entender cada compuerta como un filtro dinámico que decide qué información es relevante en cada instante.

- La ***puerta de olvido*** actúa como un regulador que atenúa recuerdos obsoletos, lo que evita que la red "se atasque" en patrones antiguos.
- La ***puerta de entrada*** permite incorporar información nueva, representando la capacidad de aprendizaje a corto plazo.
- Finalmente, la ***puerta de salida*** controla qué parte del conocimiento acumulado influirá en la predicción actual.

A.6. Otros aspectos de las redes LSTM

En el contexto financiero, resulta especialmente relevante destacar que las redes LSTM no sólo capturan dependencias temporales de un mismo activo, sino que también pueden incorporar las relaciones cruzadas o correlaciones existentes entre distintos activos, siempre que la arquitectura de la red lo permita.

Cuando se diseña una LSTM multivariante o con múltiples cabezas de salida, cada una de ellas puede especializarse en predecir el comportamiento de un activo concreto, compartiendo al mismo tiempo una parte común del procesamiento de la información. Este diseño permite que la red aprenda representaciones subyacentes que recogen la interacción entre los distintos activos (por ejemplo, detectando cambios en un sector o geografía concreta), de modo que las predicciones finales puedan reflejar no sólo la dinámica individual de cada serie, sino también su estructura de correlaciones.

De este modo, las LSTM pueden entenderse como un marco flexible para modelar dependencias tanto *intra-temporales* (a lo largo del tiempo) como *inter-activos* (entre series), ampliando considerablemente su alcance respecto a los modelos univariantes tradicionales.

Apéndice B. Arquitecturas y métricas de evaluación usadas

B.1. LSTM de retornos

B.1.1. Escalado de datos

El escalador que se usa en el modelo es el `RobustScaler()` de la librería `sklearn.preprocessing`, basado en escalar por la mediana y el rango intercuartílico:

$$x' = \frac{x - Mediana(x)}{RIC}$$

donde

$$RIC = Q_{0,75} - Q_{0,25}$$

Este escalador se denomina *robusto* porque es el que mejor se blinda frente al ruido de *outliers* en comparación con otros como el `StandardScaler()`, que asume además cierta normalidad; o el `MinMaxScaler()`, que aunque útil en modelos con redes que usen funciones de activación sigmoidales, no es óptimo en la gestión de valores extremos.

$$\textbf{StandardScaler: } x' = \frac{x - \mu}{\sigma} \qquad \textbf{MinMaxScaler: } x' = \frac{x - x_{\text{mín}}}{x_{\text{máx}} - x_{\text{mín}}}\,(b - a) + a$$

Dado que la idea es construir un modelo que sea robusto y regular en condiciones normales de mercado, sacrificando algo de eficacia en periodos de crisis, el `RobustScaler()` resulta idóneo. El escalado se hace a los datos predictores y a la variable objetivo, se entrena el modelo con ellos y las predicciones se desescalan para interpretar los resultados en su magnitud original.

B.1.2. Arquitectura y entrenamiento

Este es el paso más técnico: **definir el modelo LSTM correctamente**. En este sentido, las librerías de `Keras` y `TensorFlow` proporcionan el marco computacional necesario, de forma que únicamente se deben concretar los parámetros. Se puede resumir la especificación inicial de la arquitectura en el siguiente cuadro:

Cuadro 5: Arquitectura de la red LSTM de retornos. Fuente: Elaboración propia.

Capa	Tipo	Descripción	Dimensión de salida	Comentarios
1	Entrada	Secuencia temporal de retornos	(lookback = 60, 11)	$N = 11$: número de activos

2	LSTM	`units` = 64, `return_sequences`= `True`, regularización L2 (10^{-5})	(lookback = 60, 64)	Captura dependencias temporales
3	Dropout	`rate` = 0,2	(lookback = 60, 64)	Reduce sobreajuste
4	LSTM	`units` = 32, regularización L2 (10^{-5})	(32)	Resume la secuencia en un vector final
5	Dense	`units` = 11	(11)	Predicción multivariante (uno por activo)

Configuración

Función de pérdida	Huber ($\delta = 1$)	Robusta frente a valores extremos
Optimizador	Adam (learning_rate configurado)	Ajuste adaptativo
Regularización adicional	Dropout + L2	Control del sobreajuste

Configuración

`EarlyStopping`	Supervisar la pérdida de validación	Evita sobreajuste y reduce tiempo computacional
`ReduceLROnPlateau`	Reduce la tasa de aprendizaje cuando la métrica deja de mejorar.	Convergencia más fina y evita estancamiento
`ModelCheckpoint`	Guarda el modelo con mejor desempeño	Garantiza que el modelo final usado es el óptimo observado

La elección de estos parámetros se basa en configuraciones ampliamente utilizadas. El número de neuronas tiene que ser superior al número de activos, mientras que la tasa de aprendizaje (*learning rate*) debe seleccionarse cuidadosamente: valores demasiado altos pueden inducir subajuste, y valores demasiado bajos, sobreajuste. En la práctica es un ajuste iterativo, comenzando con una configuración estándar y ajustando posteriormente los hiperparámetros según los resultados obtenidos.

B.1.3. Métricas de evaluación

Una vez definido, el modelo se ajusta a través del comando `.fit` y, tras el tiempo de entrenamiento correspondiente, se obtiene el modelo final. A continuación, se generan las predicciones con `model.predict` y se desescalan los resultados con `.inverse_transform` para hacer la evaluación con las métricas apropiadas. En el cuadro 7 se pueden ver las métricas seleccionadas para evaluar el desempeño de esta primera red.

Cuadro 7: Métricas empleadas para evaluar el modelo. Fuente: Elaboración propia.

Nombre	Fórmula	Justificación
RMSE (Root Mean Squared Error)	$\text{RMSE} = \sqrt{\text{MSE}}$	Permite interpretar el error en las mismas unidades que la variable objetivo.
MAE (Mean Absolute Error)	$\text{MAE} = \frac{1}{TN}\sum_{t=1}^{T}\sum_{j=1}^{N}\lvert\hat{y}_{t,j} - y_{t,j}\rvert$	Menos sensible a valores extremos; mide el error medio absoluto.
MAE por activo	$\text{MAE}_j = \frac{1}{T}\sum_{t=1}^{T}\lvert\hat{y}_{t,j} - y_{t,j}\rvert$	Permite analizar el rendimiento por activo, identificando los más difíciles de predecir.
Hit Ratio (Acierto de signo)	$\text{HR} = \frac{1}{TN}\sum_{t=1}^{T}\sum_{j=1}^{N}\mathbf{1}\{\text{sign}(\hat{y}_{t,j}) = \text{sign}(y_{t,j})\}$	Evalúa si el modelo acierta la dirección del movimiento; clave cuando importa el signo más que el valor exacto.

B.1.4. Problemas de desajuste del modelo

En una primera instancia el modelo alcanzó un *Hit Ratio* de un 53 %. Aunque ligeramente superior al azar, este resultado resulta claramente insatisfactorio. Frecuentemente, las métricas de evaluación son buenos indicadores, pero pueden terminar representando entes abstractos que no señalan los problemas reales. Un *Hit Ratio* cercano al de un evento aleatorio indica que algo no está funcionando correctamente. Pero estas métricas no evidencian, en ningún caso, el error en la concepción del diseño, y eso es un problema que dificulta su mejora.

Aun así, se decidió continuar hasta completar el proceso, ya que el *backtest* es la validación empírica relevante del sistema.

Al realizar las primeras pruebas, se observó un patrón: la red tendía a producir predicciones con **baja varianza** y un **sesgo de signo** relativamente estable por activo (positivo en algunos casos y negativo en otros). Esto apunta a que la red no capturaba la dinámica temporal, sino que convergía a una predicción casi constante cuyo signo acababa dependiendo del activo. La hipótesis principal no fue un problema de hiperparámetros, sino de preprocesado de la variable objetivo.

En este contexto, Prater et al. (2024) señala el problema del sesgo hacia el valor medio que a veces sufren las redes si no están bien calibradas o la información proporcionada no es suficiente.

Inicialmente, se revisó el escalado de la variable objetivo. En particular, la documentación de scikit-learn developers (2023) contempla la opción de desactivar el centrado mediante el parámetro `with_centering=False`, de modo que no se resta la mediana y únicamente se normaliza por el rango intercuartílico. Este ajuste es relevante porque, en retornos financieros, la mediana en entrenamiento puede ser ligeramente negativa (series con deriva moderada a la baja). En ese caso, un valor cercano a 0 en el espacio escalado se mapea, tras invertir el escalado, a valores próximos a dicha mediana negativa, introduciendo un sesgo sistemático de signo. Desactivar el centrado reduce este mecanismo: si el modelo colapsa hacia valores cercanos a 0 en el espacio normalizado, el reescalado deja de trasladar automáticamente esas predicciones a la mediana histórica (y, por tanto, a un signo sesgado).

Adicionalmente, y dado que las series financieras pueden presentar deriva y no estacionariedad, se exploró la opción detraer una componente de tendencia de la variable. En línea con recomendaciones habituales para evitar que el preprocesado incorpore información inadecuada en series temporales, se aplicó una media móvil exponencial, de forma que antes del escalado se definió:

$$\bar{r_t} = r_t - EMA_{t-1}$$

donde EMA_{t-1} representa la media móvil exponencial de los últimos 60 días.

Tras estos cambios, al menos, se logró eliminar el sesgo de signo y el desempeño de la red consiguió subir hasta un *hit ratio* de en torno a 70 %. Aunque no supone una demostración causal y podrían existir soluciones más precisas y sofisticadas, la evidencia sugiere que la intervención fue satisfactoria para los objetivos de este trabajo.

B.2. LSTM de riesgos

B.2.1. Arquitectura y entrenamiento.

La especificación del modelo sigue la línea estándar de la red de retornos, cambiando la función de pérdida por la NLL. En el cuadro 9 se muestra cómo resulta la especificación de esta segunda red:

Cuadro 9: Arquitectura de la red LSTM de riesgos. Fuente: Elaboración propia.

Capa	**Tipo**	**Descripción**	**Dimensión de salida**	**Comentarios**
1	Entrada	Secuencia temporal de retornos	(lookback = 240, N)	N: número de activos
2	LSTM	`units` = 64, `return_sequences`= `True`, L2 (10^{-5})	(lookback = 240, 64)	Dependencias temporales
3	Dropout	`rate` = 0,2	(lookback = 240, 64)	Reduce sobreajuste
4	LSTM	`units` = 32, L2 (10^{-5})	(32)	Resumen de la secuencia
5	Dense	`units` = N, *sin activación*	(N)	*Logits* de varianza por activo
Optimización y entrenamiento				
Función de pérdida		NLL (Gauss), con $\hat{v} = \text{softplus}(z) + \varepsilon$		Modelado explícito de varianza
Optimizador		Adam (learning_rate configurado)		Ajuste adaptativo
Regularización adicional		Dropout + L2		Control del sobreajuste
Callbacks				
`EarlyStopping`		Monitorea `val_loss`, restaura mejores pesos		Generalización
`ReduceLROnPlateau`		Factor 0,5, paciencia moderada		Convergencia fina
`ModelCheckpoint`		Guarda el mejor modelo		Reproducibilidad

Un aspecto técnico importante es que el modelo no produce $\hat{v}$ directamente, sino un conjunto de *logits* z que se transforman mediante la función

$$\hat{v} = \text{softplus}(z) + \varepsilon$$

con ε pequeño para evitar problemas numéricos. Este mecanismo garantiza la **positividad** de la varianza sin necesidad de imponer restricciones artificiales en la arquitectura.

B.2.2. Métricas de evaluación

Para evaluar el desempeño en test se consideran dos dimensiones complementarias: precisión de nivel y *calibración*. Como métrica de nivel, se da el **QLIKE** medio y el MAE entre $|r|$ y $\hat{\sigma}$ tanto agregado como por activo:

$$\text{QLIKE} \;=\; \log(\hat{v}) + \frac{r^2}{\hat{v}}, \qquad \text{MAE}(|r|, \hat{\sigma}) \;=\; \frac{1}{TN} \sum_{t,j} \Big| |r_{t,j}| - \hat{\sigma}_{t,j} \Big|.$$

En síntesis, se mantiene y en escala real, se modela $\hat{v}$ explícitamente vía NLL y `softplus`, y se validan tanto precisión como calibración frente a una referencia estándar. Configuración estándar, reproducible y adecuada para analizar el valor incremental de la LSTM en el modelado de riesgo.

Apéndice C. Mecanismos financieros del modelo

En esta sección se detalla el funcionamiento de las cuatro opciones económicas que componen el perfil de riesgo del usuario dentro del modelo completo.

C.1. Limitación de pesos y activos

La **limitación de pesos** es fundamental cuando se quiera cubrir del riesgo intrínseco de un activo. Aunque el activo i ofrezca una rentabilidad ajustada al riesgo superior a la del resto, concentrar toda la inversión en un único activo sería excesivamente arriesgado. Por ello, se impone un límite máximo a los pesos de la cartera. En este trabajo, se ha establecido $w_{max} = 0,3$; de modo que, si la asignación sugerida por el modelo excede este valor, la señal sobrante se mantiene como posición en efectivo.

Por otra parte, la **limitación de activos** evita que el modelo diluya todo el capital invertido entre muchas opciones distintas. En un escenario hipotético con un universo de 100 activos, podrían generarse pesos mínimos como 0,001, que resultan prácticamente irrelevantes. Por este motivo, se establece un límite k al número de acciones diferentes que la cartera puede incluir. Así, el modelo calcula las señales para todos los activos y selecciona únicamente los activos más relevantes, asignando el exceso a efectivo.

C.2. Objetivo máximo de volatilidad

Los **objetivos de volatilidad** (en inglés, *volatility targeting*) limitan el riesgo global al que se expone la cartera y resultan fundamentales para mantener una estrategia segura y consistente a largo plazo. De forma análoga a lo planteado por Markowitz (1952) en su modelo de media-varianza, este enfoque busca aprovechar también las **correlaciones negativas** entre activos para reducir el riesgo total de la cartera, incluso cuando los activos individuales puedan presentar volatilidades elevadas.

Este mecanismo constituye asimismo una forma de **diversificación efectiva**: a mayor independencia sectorial entre activos, menor será la volatilidad conjunta de la cartera. El procedimiento que se realiza es el siguiente:

1. **Construcción de la matriz Varianza-Covarianza Σ.**
   Se parte de las volatilidades individuales predichas, que conforman la diagonal de la matriz, y se calculan las correlaciones entre activos en una ventana temporal de longitud $L = 240$ días.

   Recordando que:
   $$\mathrm{Cov}(A, B) = \sigma_A \sigma_B \cdot \mathrm{Corr}(A, B),$$
   la matriz de covarianza completa se obtiene como:
   $$\Sigma = D(\sigma)\,\mathrm{Corr}\,D(\sigma),$$
   donde $D(\sigma)$ denota la matriz diagonal cuyos elementos son las volatilidades marginales.

2. **Evaluación del riesgo de la cartera.**
   Una vez construida la matriz Σ, la volatilidad total de la cartera viene dada por:

$$\text{Vol}(w) = \sqrt{w^\top \Sigma\, w},$$

   siendo w el vector de pesos de la cartera.

3. **Ajuste al objetivo de volatilidad (Volatility Targeting).**
   Si la volatilidad obtenida no coincide con la volatilidad objetivo $\sigma*$, se rescalan los pesos manteniendo su proporción relativa:

$$\bar{w} = w \cdot \frac{\sigma*}{Vol(w)}$$

   Este paso permite controlar la exposición total al riesgo sin modificar la estructura interna de la cartera. Es decir, *se deciden primero qué activos se quieren en cartera, y después cuánto riesgo se busca asumir.*

En síntesis, el *volatility targeting* permite desacoplar la selección de activos de la intensidad del riesgo asumido, obteniendo una evolución temporal de la volatilidad más estable y robusta frente a cambios bruscos en el mercado.

C.3. Estrategias de cortos

En la implementación, el modelo tiene un parámetro de `short_allowed` que por defecto está en `False`. Este parámetro le indica al sistema si puede ponerse en corto. En caso de no permitirlo, todos los valores predichos como negativos se ponen a $-\infty$ indicando que dichos activos no deben incluirse en la cartera. Por el contrario, si se permitieran posiciones en corto, se toma el valor absoluto $|.|$ de la predicción $\hat{\mu}$ y se hace el *ranking* por valores mayores. Posteriormente, se seleccionan los k activos de mayor variación y se emiten las señales, guardando el signo correspondiente. Se toma:

$$w_i = sign(\hat{\mu}_i) \cdot \frac{exp(\tau s_i)}{\sum_{j=1}^{N} exp(\tau s_j)}$$

obteniéndose un vector de pesos con posiciones en corto también. Después se aplica la limitación de pesos si corresponde o el *volatility targeting*.

C.4. Apalancamiento y dinero en efectivo

Apalancamiento. A través de la opción `leverage_allowed=True` se le estará diciendo al modelo que puede apalancarse. El funcionamiento es el siguiente: si en el modelo la cartera elegida no genera volatilidad suficiente como para sobrepasar el objetivo anual, entonces se apalanca hasta llegar a este objetivo. En la evaluación de rendimientos, este apalancamiento luego se paga a un interés r_l. Por el contrario, cuando el modelo no tiene permitido apalancarse, los límites de exposición (peso máximo y número de activos) indican que el sistema debe guardar **efectivo**, que se pagará al tipo de interés libre de riesgo r_f.

Apéndice D. Resultados de las LSTM

En esta sección se presentarán las métricas de desempeño que obtuvieron las dos redes neuronales, y que han sido usadas para validación interna exclusivamente. Es decir, estas métricas no son métricas financieras o del modelo económico en sí, pero sirven también para contextualizar la validez del mismo puesto que, a fin de cuentas, las LSTM son los motores del meta-modelo global.

D.1. LSTM de retornos

Se analizan las métricas globales usuales del modelo:

RMSE	MAE	Hit Ratio	Baseline (Mediana)
0.0112	0.0083	69,64	Sí

De un lado, el RMSE de 0.0112 sería un error razonable si suponemos un mercado que habitualmente oscila entre el 0.5 y el 1 %. Asimismo, un error absoluto medio del orden de 0.8 % es coherente con la magnitud del ruido del mercado. Si bien estos resultados no son perfectos, sí son consistentes, razonables y suficientes en el contexto del mercado analizado. Por otro lado, un *hit ratio* de casi el 70 % es una señal muy positiva, ya que indica que el modelo tiene una gran capacidad direccional en los activos que se ha entrenado, acertando 7 de cada 10 direcciones. Por último el *baseline* con el que se comparaba era un modelo muy básico que devolvía la mediana de los datos, y que se usó sobre todo en el momento en que la red colapsaba para revisar su estado.

En cuanto al error absoluto medio por activo se tiene:

Activo	**MAE**
ENEL.MI	0.007
NOVN.SW	0.007
SAN.MC	0.012
SIE.DE	0.012
XLB	0.007
XLE	0.009
XLF	0.006
XLI	0.007
XLK	0.009
XLV	0.006
XLY	0.009

Cuadro 11: Error absoluto medio (MAE) por activo en el conjunto de test.

Esto muestra que Santander y Siemens tienen peor desempeño en el modelo y posiblemente vaya en línea con que no han tenido una estructura general alcista en el periodo de entrenamiento, como sí la tuvieron el resto de activos. De esta forma el modelo tiende a

sesgarse positivamente (como se explica detalladamente en la sección B.1.4 del Apéndice B) y por ello falla más con estos dos valores. En cualquier caso, no resultan errores que comprometan la fiabilidad del modelo.

D.2. LSTM de riesgos

Se estudian a continuación las métricas en la LSTM de riesgos:

QLIKE	QLIKE (EWMA)	MAE
-7.83	-4.97	0.007

El QLIKE indica la calidad en la predicción, pero no tiene significado por sí mismo, por lo que debe compararse con un *baseline*, que en este caso se usa una media móvil exponencial (se da más importancia a la observación presente que a la pasada). Como el QLIKE es menor, se puede afirmar que, en este sentido, presenta un mejor desempeño que la EWMA.

Activo	**MAE**
ENEL.MI	0.007888
NOVN.SW	0.006361
SAN.MC	0.010447
SIE.DE	0.009059
XLB	0.005635
XLE	0.008507
XLF	0.006045
XLI	0.005228
XLK	0.007199
XLV	0.005145
XLY	0.005980

Cuadro 12: Error absoluto medio (MAE) por activo para la red de riesgos.

En cuanto al MAE por activo, se observa el mismo patrón que antes, con un desempeño peor en el Santander especialmente y en Siemens. Posiblemente debido a una mayor volatilidad o a un comportamiento anómalo respecto del resto de activos.

En síntesis, se han obtenido unos resultados en las LSTM muy aceptables, que sientan una base bastante sólida sobre la que desarrollar el modelo económico previsto. Seguidamente se analizan los resultados de este.

Apéndice E. Reconversión de retornos a precios

Cabe señalar un aspecto técnico relevante: al reconvertir retornos logarítmicos en precios de activos, hay que exponenciar, ajustando el desfase que se genera para no multiplicar errores que no deben magnificarse. De lo contrario, se estaría haciendo una conversión errónea. En términos generales, el proceso de conversión acometido se basa en:

Definición de retorno logarítmico

$$g_t \;=\; \log\!\left(\frac{P_t}{P_{t-1}}\right)$$

Paso de log-retorno a factor de crecimiento y retorno aritmético

$$\frac{P_t}{P_{t-1}} \;=\; e^{g_t}$$

$$r_t \;=\; \frac{P_t - P_{t-1}}{P_{t-1}} \;=\; e^{g_t} - 1$$

Reconstrucción de precios (integración multiplicativa)

$$P_t \;=\; P_{t_0-1}\,\exp\!\left(\sum_{k=t_0}^{t} g_k\right)$$

Se reconstruye primero el log-retorno"completo" como suma de residual + componente deriva

$$\widehat{\mu}_t \;=\; \widehat{\mathrm{res}}_t \;+\; b_{t-1}$$

Reconstrucción de precios usando los log-retornos completos predichos

$$\widehat{P}_t \;=\; P_{t_0-1}\,\exp\!\left(\sum_{k=t_0}^{t} \widehat{\mu}_k\right)$$

Calibración de sesgo (global) sobre los log-retornos predichos Sea $t \in \mathcal{T}$ el conjunto de fechas usadas para calibrar.

$$\delta \;=\; \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} (\widehat{\mu}_t - g_t)$$

$$\widehat{\mu}_t^{\,\mathrm{cal}} \;=\; \widehat{\mu}_t - \delta$$

Re-anclaje periódico para evitar deriva: Sea τ el último día observado real antes de iniciar el tramo (o antes de cada bloque).

$$\widehat{P}_t^{\,(\tau)} \;=\; P_\tau\,\exp\!\left(\sum_{k=\tau+1}^{t} \widehat{\mu}_k^{\,\mathrm{cal}}\right)$$

Si no se realizara este proceso, los pequeños errores de conversión se exponencian constantemente, dando lugar a errores que no se corresponden con la realidad.

Apéndice F. Backtest apalancado

Esta sección presenta un backtest donde el parámetro de apalancamiento se ha activado, además se hace una pequeña prueba donde el modelo tiene que lidiar con costes para ver cómo rinde.

Así, se le ha indicado al sistema que, además de tomar posiciones en corto, puede apalancarse siguiendo las reglas que se han establecido. Lo que cabe esperar es que el modelo obtenga más ganancias en general, pero que no haya cambios ni de volatilidad ni mejoras en VaR. En otras palabras, que el apalancamiento actúe como impulso de los retornos, pero no como instrumento amortiguador.

El sistema inicialmente no incluye costes de endeudamiento, es decir, se asume la posibilidad de apalancamiento sin coste alguno. En la figura 18 se ve que, en ausencia de costes de endeudamiento, esta resulta la estrategia vencedora.

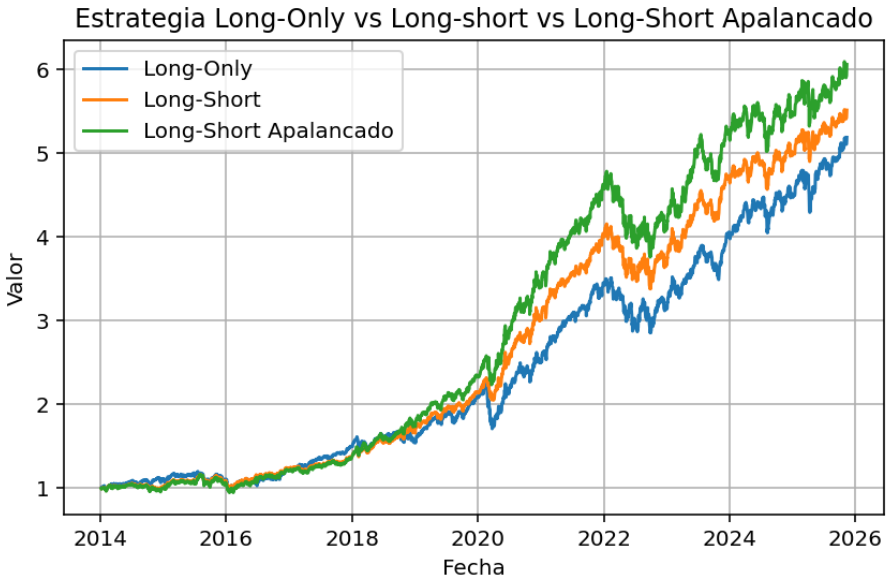

Figura 18: Comparativa de las diferentes estrategias (sin costes de endeudamiento). Fuente: Elaboración propia.

Ahora bien, en una pequeña prueba con un coste de endeudamiento del 2 % (TAE), y recordando que el modelo no atiende a costes de transacción, su desempeño se deteriora notablemente, tal y como se observa en la figura 19.

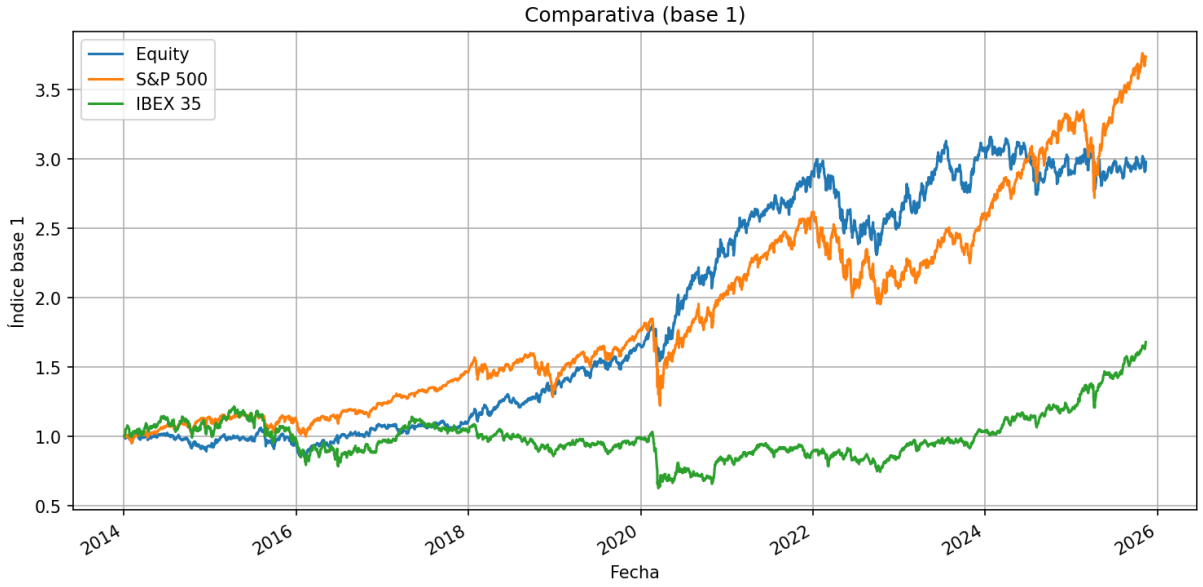

Figura 19: Evolución de la cartera apalancada con costes de endeudamiento. Fuente: Elaboración propia.

Apéndice G. Cuadro de hiperparámetros de la red

Cuadro 13: Hiperparámetros y valores por defecto. Fuente: Elaboración propia.

Hiperparámetro	**Valor por defecto**	**Descripción Breve**
Descarga		
tickers	{XLV, XLF, . . . , SAN.MC} = 11	Activos con los que se entrena el modelo
winsor_lower	0,005	Percentil inferior para winsorización de retornos
winsor_upper	0,995	Percentil superior para winsorización de retornos
start_date	2013-01-01	Fecha inicial de descarga (por defecto)
LSTM de retornos		
lookback	60	Ventana de entrada
horizon	1	Pasos adelante
train_split	0,7	Ratio entrenamiento
val_split	0,15	Ratio validación
neurons1	64	Unidades primera LSTM
neurons2	32	Unidades segunda LSTM
dropout	0,2	*dropout*
learning_rate	0,001	*learning rate* Adam
epochs	200	Épocas de entrenamiento
batch_size	32	Tamaño de *batch*
early_stopping_patience	15	Paciencia ES
reduce_lr_factor	0,5	Factor reduce LR

reduce_lr_patience	7	Paciencia reduce LR
reduce_lr_min_lr	0,00001	Mínimo LR
ewma_span	20	Suavizado base EWMA retornos
X_scaler	`RobustScaler()`	Escalado de entradas
y_scaler	`RobustScaler(with_centering=False)`	Escalado de salidas
LSTM de riesgos		
lookback	240	Ventana de entrada
horizon	1	Horizonte
neurons1	64	Unidades primera LSTM
neurons2	32	Unidades segunda LSTM
dropout	0,2	*dropout*
learning_rate	0,001	*learning rate* Adam
epochs	200	Épocas
batch_size	32	Tamaño *batch*
var_softplus_eps	1E-29	ϵ estabilidad varianza
ewma_sigma_lambda	0,97	λ EWMA diagnóstico σ
Cartera		
tau	6	Temperatura softmax top-k
w_max	0,3	Límite peso por activo (en *wrapper*)
permitir_cortos	FALSO	Permitir cortos
annual_vol_target	0.15	Límite de volatilidad anualizada
Backtest		
k	4	Número de activos seleccionados

rf	0	Rentabilidad libre de riesgo
alpha_VaR	0,01	Nivel VaR para test

Apéndice H. Códigos utilizados

Como ya se ha señalado, los códigos han sido hechos en Python gracias a múltiples librerías como Pandas, Numpy, Tensorflow o Keras (entre otras). Dada la extensión de los *scripts*, se adjunta una URL a un repositorio **privado** en GitHub. En él se encuentra un archivo *README* con las instrucciones para instalar y ejecutar el código.

El enlace al repositorio es el siguiente:

https://github.com/jmolinero40/Redes-LSTM-en-predicci-n-financiera.git